EX1048

48 チャンネル 高精度熱電対計測器

取扱説明書

P/N: 82-0105-001
改訂**: 2009** 年 **2** 月 **6** 日

VTI Instruments Corp.

2031 Main Street
Irvine, CA 92614-6509
(949) 955-1894

目次

はじめに

本製品の認可 7
製品保証のご案内 7
保証対象外事項 7
制限付権利に関する表示 7
安全上のご注意 8
安全表示 8
安全上の注意事項 8
サポート情報 10

第 1 章 11

製品の紹介 11
概要 11
特徴 12
チャンネルの独立性 12
測定範囲 12
ハードウェア・フィルタ 12
熱電対断線(OTC)検出 12
冷接点補償(CJC) 13
入力端子 13
自己校正 13
サンプリングレート 13
デジタル I/Oと限界値 13
LXIトリガバス 14
トリガ 14
内蔵ウェブインターフェイス 14
EX1048 仕様表 15
仕様の説明 17
測定性能の最適化 18
自己校正の使用 18
冷接点を熱安定させる 18
ハードウェアフィルタの選択 20
サンプリングレートの選択 20
設置場所の選択 20
熱電対線の選択 20

第 2 章 21

準備 21
製品の構成 21
設置場所 21
ウォーミングアップ 21
ネットワーク設定 21
時間設定 23
入力接点・配線 23

第 3 章 25

基本操作 25

はじめに 25
工学単位(EU)変換 25
ハードウェアフィルタ 26
測定範囲・入力保護 26
冷接点補償(CJC) 27
温度単位 27
サンプリングレート・ノイズ性能 28
スキャンリスト設定 28
スキャンリストタイミング 29
自己校正 29
熱電対断線検出・限界値 30
デジタルI/OとDIOリミットイベント 31
LXIトリガバス 33
ロック 35
トリガ 35
データ形式 35
データ収集 35
データ読み出し 36
ユーザー定義変換式 37
第 4 章 39
トリガ 39
はじめに 39
イベント 40
測定性能の最適化 41
最高トリガレート 42
第 5 章 43
ウェブインターフェイス操作 43
はじめに 43
ウェブインターフェイスを開く 43
基本ウェブ操作 43
Trigger（トリガ）メニュー 44
Configuration（設定） 44
Initialize（開始） 44
Abort（中止） 44
Software Arm（ソフトウェア・アーム） 45
Software Trigger（ソフトウェア・トリガ） 45
Scan List（スキャンリスト）メニュー 45
Configuration（設定） 45
Filters（フィルタ） 46
Data（データ）メニュー 46
Get FIFO Count（FIFOカウント取得） 46
Retrieve Data（データ読み出し） 46
Continuous Polling（連続ポーリング） 46
Download As File（ファイルとしてダウンロード） 46
FIFO Configuration（FIFO設定） 46
IO（入出力）メニュー 47
Digital IO（デジタルI/O） 47
Trigger Bus（トリガバス） 48
Limits（限界値）メニュー 48

Limits（限界値） 48
DIO Limit Events（DIOリミットイベント） 49
Device（デバイス）メニュー 50
Self Calibration（自己校正） 51
Network Configuration（ネットワーク設定） 52
Time Configuration（時間設定） 53
Upgrade Firmware（ファームウェアの更新） 53
Reset Device（デバイスリセット） 54
Hard Reboot（再起動） 54
Locking（ロック）メニュー 54
Lock（ロック） 54
Unlock（ロック解除） 54
Break Lock（ロックの切断） 55
Advanced Menu（高度設定） 55
User Conversions（ユーザー定義変換） 55
User CJC Temp（ユーザーCJC温度） 55
設定例 55
例１ 55
例２ 56
第 6 章 59
プログラミング 59
はじめに 59
初期設定 59
基本命令・クエリ 60
デバイスの初期化 60
デバイスのリセット 60
自己校正の実行 61
ロックの取得 62
プログラミング・シーケンス 63
測定命令・クエリ 64
スキャンリストの設定 64
工学単位変換係数の設定 64
高度変換設定 64
フィルタ周波数の設定 66
FIFOの設定 66
限界値システムの設定 67
デジタル I/Oシステムの設定 68
トリガソースの設定 70
トリガモデルの施行 74
トリガバスの設定 75
トリガイベント 76
収集データの読み出し 77
プログラム例 80
第 7 章 85
命令一覧 85
はじめに 85
アルファベット順命令リスト 85
命令一覧 88
vtex1048_abort 89
vtex1048_break_lock 90

vtex1048_check_lock....91
vtex1048_close....92
vtex1048_get_accum_limit_status....93
vtex1048_get_arm_count....94
vtex1048_get_arm_delay....95
vtex1048_get_arm_infinite....96
vtex1048_get_arm_source....97
vtex1048_get_channel_conversion....98
vtex1048_get_dio_input....99
vtex1048_get_dio_limit_event....100
vtex1048_get_dio_limit_event_invert....101
vtex1048_get_dio_limit_event_latch....102
vtex1048_get_dio_output....103
vtex1048_get_dio_output_enable....104
vtex1048_get_fifo_config....105
vtex1048_get_fifo_count....106
vtex1048_get_filt_freq....107
vtex1048_get_init_cont....108
vtex1048_get_limit_set0....109
vtex1048_get_limit_set0_manual....110
vtex1048_get_limit_set1....111
vtex1048_get_scanlist....112
vtex1048_get_trigger_count....113
vtex1048_get_trigger_delay....114
vtex1048_get_trigger_infinite....115
vtex1048_get_trigger_source....116
vtex1048_get_trigger_timer....117
vtex1048_get_user_cjc_enable....118
vtex1048_get_user_cjc_temp....119
vtex1048_get_user_conversion....120
vtex1048_get_vtb_input....121
vtex1048_get_vtb_output....122
vtex1048_get_vtb_output_enable....123
vtex1048_init....124
vtex1048_init_imm....125
vtex1048_lock....126
vtex1048_read_fifo....127
vtex1048_reset....128
vtex1048_reset_fifo....129
vtex1048_reset_trigger_arm....130
vtex1048_revisionQuery....131
vtex1048_self_cal_clear....132
vtex1048_self_cal_clear_stored....133
vtex1048_self_cal_get_status....134
vtex1048_self_cal_init....135
vtex1048_self_cal_is_stored....136
vtex1048_self_cal_load....137
vtex1048_self_cal_store....138
vtex1048_set_arm_count....139
vtex1048_set_arm_delay....140
vtex1048_set_arm_infinite....141
vtex1048_set_arm_source....142
vtex1048_set_channel_conversion....143
vtex1048_set_dio_limit_event....144
vtex1048_set_dio_limit_event_invert....145
vtex1048_set_dio_limit_event_latch....146

vtex1048_set_dio_output 147
vtex1048_set_dio_output_enable 148
vtex1048_set_dio_pulse 149
vtex1048_set_fifo_config 150
vtex1048_set_filt_freq 151
vtex1048_set_init_cont 152
vtex1048_set_limit_set0 153
vtex1048_set_limit_set0_manual 154
vtex1048_set_limit_set1 155
vtex1048_set_scanlist 156
vtex1048_set_trig_source_timer 157
vtex1048_set_trigger_count 158
vtex1048_set_trigger_delay 159
vtex1048_set_trigger_infinite 160
vtex1048_set_trigger_source 161
vtex1048_set_trigger_timer 162
vtex1048_set_user_cjc_enable 163
vtex1048_set_user_cjc_temp 164
vtex1048_set_user_conversion 165
vtex1048_set_vtb_output 166
vtex1048_set_vtb_output_enable 167
vtex1048_set_vtb_pulse 168
vtex1048_soft_arm 169
vtex1048_soft_trigger 170
vtex1048_unlock 171

第 8 章 173

動作理論 173
はじめに 173
シグナルコンディション回路 173
冷接点補償 174
校正 174
熱電対計算 174

総索引 175

本製品の認可

VTI Instruments Corp. (VTI) certifies that this product met its published specifications at the time of shipment from the factory. VTI further certifies that its calibration measurements are traceable to the United States National Institute of Standards and Technology (formerly National Bureau of Standards), to the extent allowed by that organization's calibration facility, and to the calibration facilities of other International Standards Organization members.

製品保証のご案内

The product referred to herein is warranted against defects in material and workmanship for a period of three years from the receipt date of the product at customer's facility. The sole and exclusive remedy for breach of any warranty concerning these goods shall be repair or replacement of defective parts, or a refund of the purchase price, to be determined at the option of VTI.

For warranty service or repair, this product must be returned to a VTI Instruments authorized service center. The product shall be shipped prepaid to VTI and VTI shall prepay all returns of the product to the buyer. However, the buyer shall pay all shipping charges, duties, and taxes for products returned to VTI from another country.

VTI warrants that its software and firmware designated by VTI for use with a product will execute its programming when properly installed on that product. VTI does not however warrant that the operation of the product, or software, or firmware will be uninterrupted or error free.

保証対象外事項

The warranty shall not apply to defects resulting from improper or inadequate maintenance by the buyer, buyer-supplied products or interfacing, unauthorized modification or misuse, operation outside the environmental specifications for the product, or improper site preparation or maintenance.

VTI Instruments Corp. shall not be liable for injury to property other than the goods themselves. Other than the limited warranty stated above, VTI Instruments Corp. makes no other warranties, express or implied, with respect to the quality of product beyond the description of the goods on the face of the contract. VTI specifically disclaims the implied warranties of merchantability and fitness for a particular purpose.

制限付権利に関する表示

Use, duplication, or disclosure by the Government is subject to restrictions as set forth in subdivision (b)(3)(ii) of the Rights in Technical Data and Computer Software clause in DFARS 252.227-7013.

VTI Instruments Corp.
2031 Main Street
Irvine, CA 92614-6509 U.S.A.

安全上のご注意

お使いになる方や他の方への危害及び財産への破損を未然に防止する為、必ずお守り頂きたい事項を、ここにご説明申し上げます。この製品の取り扱い及び修理全般に関しましては、これらの注意をお守りください。また、表示内容を無視して誤った方法で取り扱うことは、この製品のデザイン、生産過程、使用に於ける安全基準の範囲外にある事をここに記します。

この製品の修理は資格があるサービスエンジニアにお任せください。

安全表示

このマニュアルには次のような表示が使われています。

警告　死亡または傷害の恐れのある警告事項です。

注意　物質的損害またはデータ紛失の恐れのある注意事項です。

本製品には次のような表示が使われています。

注意－安全上の表記

アース

安全上の注意事項

傷害または物質的損害を避けるためには次のことをお守りください。

規定された電源コードの使用　危険を避けるためには指定の電源コードをお使いください。

規定された電源の使用　指定以外の電源を使うことは過負荷、感電、火災の原因になります。

規定されたヒューズの使用　規定以外のヒューズの使用は火災の原因になります。

安全上の注意事項(つづき)

感電の注意

感電、火災の原因になりますので、カバーをはずした状態では使用しないでください。電源につながった状態でのケーブル、プローブ、導線などの抜き差しはしないでください。修理を行う前には必ず電源を切り、すべてのケーブル類を抜いてください。***この製品の修理は資格があるサービスエンジニアにお任せください。***

機器のアース

本製品は電源コードのアース線を通してアースしています。感電防止のためにアース線を接地してください。

使用環境

傷害、感電、火災を防止する為：

- ぬれた、または湿気のある状態では使わないでください。
- 起爆性のある環境で使わないでください。
- 規定の温度域以外では使わないでください。
- 過熱防止のため、換気に十分なすきまをあけてください。
- 本製品が故障している恐れがある場合は絶対に使用しないでください。***この製品の点検、修理は資格があるサービスエンジニアにお任せください。***

誤ったご利用

機器がマニュアルで明記されていない使われ方をした場合、本製品の安全機能が破損する可能性があります。
本製品の安全性は、点検によって確認されます。

サポート情報

本製品のサポートはインターネット又は VTI Instruments 社お客様サポートセンターにお問い合わせください。

VTI Instruments
World Headquarters

VTI Instruments Corp.
2031 Main Street
Irvine, CA 92614-6509

電話: (949) 955-1894
ファックス: (949) 955-3041

VTI Instruments
Cleveland Instrument Division

5425 Warner Road
Suite 13
Valley View, OH 44125

電話: (216) 447-8950
ファックス: (216) 447-8951

VTI Instruments
Lake Stevens Instrument Division

VTI Instruments Corp.
1924 - 203 Bickford
Snohomish, WA 98290

電話: (425) 212-2285
ファックス: (425) 212-2289

テクニカルサポート

電話: (949) 955-1894
ファックス: (949) 955-3041
E-mail: support@vtiinstruments.com

世界各地のサポートサイト・サービスプラン情報は http://www.vtiinstruments.com でお求めください。

第 1 章

製品の紹介

概要

EX1048 は 48 チャンネル高精度熱電対計測器です。そしてその計測性能、信頼性、柔軟性、高い実装密度、ネットワークへの接続性により、その性能と使いやすさにおいて類のないものに仕上がっています。EX1048 はイーサネットを通じた通信が可能な独立した温度管理システムであり、その統合性は従来のデータ収集機器における端子ブロック、信号調節カード、デジタイザ、電源、シャーシなどの結合という複雑な作業からユーザーを解放します。

また、EX1048 は従来のデータ収集機器とは比べ物にならないレベルの計測信頼性とチャンネルの独立性をもっています。スキャン速度やチャンネルの順序、オーバーロードの影響といった心配も取り除いてくれます。EX1048 ではスキャン速度やチャンネルの状態に関わらずチャンネル同の影響はありません。また、高性能なコモンモード除去機能により電力線による干渉だけでなく、高周波ノイズの影響からも免れます。高精度な熱電対の断線検出機能を提供することによって全体的なシステムの信頼性を維持することができます。

EX1048 は又、機器構成に対して高い柔軟性を提供します。ユーザーによるチャンネル毎の測定、ハードウェア・フィルタ設定、限界値の設定に対応しています。標準的な熱電対に加え EX1048 はユーザー指定の熱電対多頂式の入力に対応、低ノイズなミリボルト計としても使えます。スキャン速度はスキャンチャンネル数に関係なく最高 1 kHz までプログラム可能です。これらのフィルタとスキャン速度の組み合わせによって EX1048 は高感度が必要な測定から高速細ゲージ熱電対などの速度重視な測定まで可能にします。

最も高レベルな精度と安定性を提供するために EX1048 は熱電対 4 チャンネルにつき 1 つ、計 12 のサーミスタによって監視された恒温入力部を備えています。さらに校正ソースを内蔵しているので命令 1 つで自己校正できます。これによって広範囲の使用環境でのご利用に対応しています。お客様にとって使いやすくする為、この取扱説明書では 15 °C から 35 °C までの環境温度における自己校正使用時、及び未使用時の温度精度仕様を提供します。

EX1048 が誇るのは、その高機能に劣らぬ使い易さでもあります。内蔵ウェブインターフェイスによって、機器の状態や通信が瞬時に、簡単に確認できます。また、業界標準の VXI*plug&play* ドライバにより、慣れ親しんだ API での開発が可能になり、システムの構築、開発時間の短縮に繋がります。

特徴

チャンネルの独立性

EX1048 が装備している 48 の差動入力チャンネルは、独立性シグナルコンディションパスであり、それぞれに増幅回路、プログラマブルハードウェアフィルタ、そして連続的熱電対断線検出回路を装備しています。この独立性により、ユーザーはマルチチャンネルデータ収集システムに多いチャンネル間の干渉から悩まされることもなくなります。EX1048 では、スキャン速度やチャンネル状況に関係なくチャンネル間の干渉が起こることはありません。つまり、チャンネルの断線、著しい過負荷が他のチャンネルの測定結果に影響を及ぼすことがなくなりました。

測定範囲

EX1048 の温度測定範囲は、入力電圧範囲と熱電対自体の能力に影響されます。一般的な熱電対における EX1048 の測定範囲は以下のとおりです。

	最低値 (°C)	最高値 (°C)	最低値 (°F)	最高値 (°F)
J 熱電対	-200	1200	-328	2192
K 熱電対	-200	1372	-328	2502
T 熱電対	-200	400	-328	752
E 熱電対	-200	900	-328	1652
S 熱電対	-50	1768	-58	3214
R 熱電対	-50	1768	-58	3214
B 熱電対	250	1820	482	3308
N 熱電対	-200	1300	-328	2372

表1-1: EX1048 の温度測定範囲

最高の柔軟性を提供する為、熱電対変換において各チャンネルは独自に設定可能です。さらに、非標準熱電対も熱電対多項式のユーザーによる係数入力に対応することによって可能にしています。

ハードウェア・フィルタ

EX1048 の入力チャンネルは、ハードウェアフィルタによりそれぞれ 4 Hz 又は 1 kHz の遮断周波数を設定することが可能です。この機能により EX1048 は 1 ユニットで多目的に利用できます。4 Hz 設定は、最低レベルのノイズフロアと高性能のコモンモード除去を提供し、大抵の用途に適しています。さらに高スピードが必要な用途には 1 kHz 設定により最も高速な細ゲージ熱電対の出力でもほとんど歪み無く測定できます。

熱電対断線(OTC)検出

EX1048 の統合性は、システムデザイナーが一般的に直面する信頼性と接続性の問題の多くを取り除いてくれますが、それでもまだセンサの接続自体の信頼性を心配しなくてはいけません。幸い、EX1048 は、この点でも大きくユーザーを支援します。本製品の各入力チャンネルは、微小な電圧によってバイアスされています。センサの接続が断たれた場合、この電流は入力アンプをアウトオブバウンド状態として示すよう駆動します。このような手法を使うことによって、入力の監視は連続的に行われ、ユーザーが別個に命令を出す必要はありません。センサの断線は据付け時だけでなくテスト中にも起こる物なので、命令でチェックするシステムより、高い信頼性を提供します。加えて、EX1048 は断線の検出と共に点灯する OTC LED を、フロントパネル部に各チャンネル分

備えます。これにより、問題箇所が簡単にすぐ判別できます。また、断続的な問題も認識されるよう、障害検出器はラッチングメカニズムにより、新しい測定シーケンスを始めるまで情報を保持します。

冷接点補償(CJC)

最も高レベルな精度と安定性を提供するために EX1048 は、熱電対 4 チャンネルにつき 1 つ、計 12 のサーミスタによって監視された等温入力部を備えています。CJC データが常に入力チャンネルと時間的に対応していることを保証するため、CJC チャンネルは毎スキャンごとに測定され、入力チャンネルの測定と対応する CJC チャンネルの測定の時間的ずれは 4 ms 以内に制限されています。外付けの冷接点を使いたい場合は、各入力チャンネルに一台、計 48 の個別な冷接点を使うことができます。また、チャンネルごとの内・外の冷接点の利用選択も可能です。

入力端子

EX1048 は、入力端子に無補償（Cu-Cu）mini-TC 雌型端子を使います。この端子により、高い信頼性を維持できると共に、取り外しも簡単になります。また、熱電対の種類に依存しないので、ハードウェアの変更なしで複数の異種熱電対を 1 台で扱うことができます。

自己校正

広い動作環境範囲において高い測定精度を提供する為、EX1048 は、自己校正の機能を備えています。自己校正時には、入力シグナルコンディションパスは入力端子の接続を断ち、内蔵校正ソースで駆動する校正バスに接続されます。複数の校正ソースポイントにおいてコンディションパスを測定することによって、各パスの電圧ゲインとオフセットが計算されます。これによって、外部装置の接続の必要なく EX1048 の測定性能を大幅に高めることができます。また、自己校正作業は、短時間で終わる上にユーザーが入力端子を切断する必要がないので、簡単に何度でも行えます。

サンプリングレート

EX1048 は、スキャンリスト上のチャンネル数に関わらず最高 1 kHz まで設定可能です。但し、サンプリングレートが 1 kHz より著しく少ない場合は EX1048 が自動的に複数のサンプルを取り、その平均値を算出します。こうすることによって、EX1048 は要求された速度でデータ出力をしながらノイズ性能を上げることができます。

デジタル I/O と限界値

EX1048 は、熱電対断線検出および汎用入力チャンネルモニタリングに使われる二種類のプログラマブル限界値を提供します。これらの限界値は、チャンネル毎にプログラム可能で、各スキャン完了時に評価されます。限界値評価の出力は、3 つの形で表示されます。フロントパネル LED の動作は、上限値、下限値の限界値一対に結び付けられています。デジタル I/O ポートの動作は、ユーザー好みの上限値、下限値の組み合わせをどちらか、または両方の限界値セットに関連付け可能です。最後に、限界値状況の情報は、機器のドライバより入手可能です。

EX1048 は、リア（背面）パネル部に 8 チャンネルデジタル I/O ポートを備えています。このポートは、アーム・トリガソース、限界値評価情報の表示、及び汎用出力デバイスとして利用可能です。汎用出力デバイスとして使う場合、各 DIO（デジタル I/O）チャンネル は、出力時の出力機能と取るべきスタティックレベルに対して独自にプログラム可能です。更に拡張、自動化された作業の為、各 DIO チャンネルは個別に 1 ないし複数の入力チャンネルに対して、1 ないし複数の限界値状態に関連付けできます。

LXI トリガバス

EX1048 は、機器リア（背面）パネル部に 8 チャンネル LXI (*LAN eXtensions for Instrumentation*)トリガバスを備えています。この差動ペア LVDS バスは、並列接続された二つの同種ポートによって構成されています。トリガバスの主用途はマルチユニットトリガ、及び同期用の高速信号伝達です。

トリガ

EX1048 は、独立したアームソース、及びトリガソースイベント構造を備えたフルファンクショントリガモデルに対応しています。トリガ、アームソースイベントは直接、タイマー、デジタル I/O、トリガバスなど様々なソースを利用して独自にプログラム可能です。

内蔵ウェブインターフェイス

EX1048 は、プログラム不要な遠隔操作を可能にするためにウェブインターフェイスを内蔵しています。ほとんどの計器に関する設定はこのウェブインターフェイスで変更可能です。また、ネットワーク設定の管理もこの画面から行います。

EX1048 仕様表

一般仕様	
チャンネル数	
	48 差動入力
関数	
	J, K, T, E, S, R, B, N, mV
サンプリングレート	
	最高チャンネル毎 1000 Sa/s
温度解像度	
	0.01 °C
温度精度	
	表 1-2 、 表 1-3 参照
温度ノイズ	
Peak-to-peak	0.08 °C 代表値 (J, K, T, E)
電圧入力域	
	±66 mV
電圧解像度	
	1 µV
電圧精度	
自己校正使用時	±(0.05% + 10 µV)
自己校正未使用時	±(0.1% + 30 µV)
電圧オフセット安定性	
	1 µV/°C 代表値
電圧ゲイン安定性	
	25 ppm/°C 代表値
入力インピーダンス	
	40 MΩ 差動入力
入力バイアス電流	
	7.5 nA 代表値
コモンモード入力域	
	±10 V
コモンモード除去比 (CMRR)	
4 Hz フィルタ	
dc	100 dB minimum
(50/60) Hz	140 dB 代表値, 120 dB minimum
1 kHz フィルタ	
dc	100 dB minimum
(50/60) Hz	100 dB 代表値, 80 dB minimum
フィルタ	
	4 Hz, 1 kHz (チャンネルごとに設定可能)
入力保護	
	±35 V
LAN	
	10/100 Base-T
入力端子	
	Cu-Cu mini-TC 端子
動作環境	
	0 °C ~ 50 °C
電源仕様	
	(90 ~ 264) V ac, (50/60) Hz, 25 VA （最大）
外形寸法	
	幅 440 mm×奥行 345 mm×高さ 45 mm

表1-2: 熱電対精度（自己校正使用時）

	-100 °C	0 °C	100 °C	300 °C	500 °C	700 °C	900 °C	1100 °C	1400 °C
J 熱電対	±0.60 °C	±0.45 °C	±0.45 °C	±0.50 °C	±0.60 °C	±0.65 °C	±0.75 °C	±0.90 °C	----------
K 熱電対	±0.70 °C	±0.50 °C	±0.50 °C	±0.60 °C	±0.65 °C	±0.75 °C	±0.90 °C	±1.10 °C	----------
T 熱電対	±0.75 °C	±0.50 °C	±0.45°C	±0.45 °C	----------	----------	----------	----------	----------
E 熱電対	±0.60 °C	±0.45 °C	±0.40 °C	±0.45 °C	±0.50 °C	±0.60 °C	±0.75 °C	----------	----------
S 熱電対	----------	±2.00 °C	±1.50 °C	±1.30 °C	±1.30 °C	±1.30 °C	±1.40 °C	±1.40 °C	±1.50 °C
R 熱電対	----------	±2.00 °C	±1.50 °C	±1.20 °C	±1.20 °C	±1.20 °C	±1.20 °C	±1.30 °C	±1.40 °C
B 熱電対	----------	----------	----------	±3.30 °C	±2.10 °C	±1.60 °C	±1.40 °C	±1.30 °C	±1.30 °C
N 熱電対	±0.80 °C	±0.60 °C	±0.55 °C	±0.55 °C	±0.60 °C	±0.70 °C	±0.80 °C	±0.95 °C	----------

Conditions[1]
Guaranteed maximum limits. Typical errors are approximately ½ of maximum.
<30 days, ±2 °C from last self-calibration
15 °C to 35 °C, 1 year from full calibration
60 minute warm-up
Exclusive of thermocouple errors
Exclusive of noise
Vcm = 0

表1-3: 熱電対精度（自己校正未使用時）

	-100 °C	0 °C	100 °C	300 °C	500 °C	700 °C	900 °C	1100 °C	1400 °C
J 熱電対	±1.00 °C	±0.75 °C	±0.75 °C	±0.90 °C	±1.10 °C	±1.20 °C	±1.40 °C	±1.70 °C	----------
K 熱電対	±1.30 °C	±0.90 °C	±0.90 °C	±1.10 °C	±1.30 °C	±1.50 °C	±1.80 °C	±2.10 °C	----------
T 熱電対	±1.30 °C	±0.90 °C	±0.80 °C	±0.80 °C	----------	----------	----------	----------	----------
E 熱電対	±1.00 °C	±0.70 °C	±0.65 °C	±0.75 °C	±0.90 °C	±1.10 °C	±1.40 °C	----------	----------
S 熱電対	----------	±4.90 °C	±3.70 °C	±3.10 °C	±3.10 °C	±3.10 °C	±3.10 °C	±3.20 °C	±3.40 °C
R 熱電対	----------	±5.00 °C	±3.60 °C	±2.90 °C	±2.80 °C	±2.80 °C	±2.80 °C	±2.80 °C	±3.00 °C
B 熱電対	----------	----------	----------	±8.50 °C	±5.30 °C	±4.10 °C	±3.50 °C	±3.20 °C	±3.10 °C
N 熱電対	±1.60 °C	±1.20 °C	±1.10 °C	±1.10 °C	±1.20 °C	±1.40 °C	±1.60 °C	±1.90 °C	----------

Conditions[1]
Guaranteed maximum limits. Typical errors are approximately ½ of maximum.
15 °C to 35 °C, 1 year from full calibration
60 minute warm-up
Exclusive of thermocouple errors
Exclusive of noise
Vcm = 0

注釈
[1] *4 Hz フィルタ設定において。1 kHz の場合は 10 µV 加算すること*

EX1048 の基本精度は 15 ºC ~ 35 ºCの使用環境において一年以内に完全校正をしていることを前提としています。この精度は、*表 1-3: 熱電対精度（自己校正未使用時）*に詳細に記されています。この基本精度は、定期的な自己校正をすることによって、大きく高めることができます。自己校正がされた場合、表 1-2 の *熱電対精度（自己校正使用時）*で記されている数値は±2 ºC までの仕様環境変化において、30 日間まで有効です。また、使用環境、完全校正の時間間隔において、基本精度の時と同じ制約が付くので、注意してください。

EX1048 の熱電対精度表は、ユーザーに対して簡単で判りやすく、機器のダイナミックレンジ内の幾つかの温度での精度を提供します。これらの表において、各熱電対タイプは横の列で表されています。各縦の列は入力温度を表しています。たとえば、*熱電対精度（自己校正使用時）*の表では、K熱電対に対し 100 ºC の温度を入力した場合、熱電対誤差とは別に最大±0.50 ºCの不確定性が含まれます。

熱電対誤差とは別に

前項で「熱電対誤差とは別に」と書いたのは、すべての熱電対に付いて回る 2 つの内在的不確定的要素を指します。まず、一つ目は、熱電対自体の精度が誤差となります。これは、各熱電対が標準の 1990 年国際温度目盛（ITS-90）に表記された特性値からのずれ。二つ目は、EX1048 の入力バイアス電流に対して電圧低下の元となる、熱電対線の抵抗です。ほとんどのアプリケーションにおいては、この誤差は、無視できるレベルです。しかし、利用される熱電対線の長さとゲージが、250 Ω 以上の抵抗値を持っている場合は、その効果も計算に入れる必要が出てきます。

例として、200 m の 24 ゲージ T 熱電対は、約 500 Ω の抵抗値を持っています。EX1048 の一般的な入力バイアス電流 7.5 nA に対して、これによる電圧誤差は、

$$500\,\Omega \times 7.5\,\text{nA} = 3.75\,\mu\text{V}$$

となります。

この誤差を、対応する温度誤差に変換する為、今度は必要な熱電対特性曲線の傾斜で割ります。この場合、T 熱電対の 0 ºC での特性曲線の傾斜は、39 µV/ºC です。故に、ここでの温度誤差は、

$$3.75\,\mu\text{V} \div 39\,\mu\text{V}/^{\circ}\text{C} = 0.1\,^{\circ}\text{C}$$

となります。

コモンモード除去

EX1048 のコモンモード除去特性は、dc、及び (50/60) Hz の干渉について dB の単位で明示されています。また、これらの仕様は、温度で無く電圧に対する測定の誤差であることに注意してください。次の例は、T 熱電対で 0 ºC の温度を測る場合の直流 1 V のコモンモード信号による温度誤差を計算します。

EX1048 は dc において最低 100 dB の CMRR（コモンモード除去比）を持っています。1 V のコモンモード電圧による最大差動電圧誤差は、

$$\frac{1\,\mathrm{V}}{10^{\left(\frac{100}{20}\right)}} = 10\,\mu\mathrm{V}$$

となります。

この電圧誤差を温度誤差に変換するには、先回と同じように適切な熱電対曲線の傾斜で割る事によって求めることができます。

$$10\,\mu\mathrm{V} \div 39\,\mu\mathrm{V/°C} = 0.26\,\mathrm{°C}$$

1 kHz フィルタ設定

最後に、1 kHzフィルタ設定における追加誤差を求める為に、同じ変換式を使います（*注釈*参照）。同じ測定ポイントにおいて、10 µVの追加誤差は、以下の温度誤差に換算されます。

$$10\,\mu\mathrm{V} \div 39\,\mu\mathrm{V/°C} = 0.26\,\mathrm{°C}$$

測定性能の最適化

ここでは、EX1048 の実性能を最大限に引き出す、またはユーザーによる一般的な過ちを避ける為のヒントと手順を紹介します。

自己校正の使用

自己校正は、可能な範囲で、なるべく頻繁に行ってください。特に、前の校正時から環境温度が大きく変わった場合は、自己校正の使用は大事になります。ただ、急な環境変化の後には、自己校正の使用は、一定の熱安定を待ってから行うことが望ましいです。自己校正作業は、短時間で終わる上、入力端子の切断や外部機器の利用が必要ないので、簡単に何度でも行えます。

冷接点を熱安定させる

熱電対計測器において、その測定精度の大部分は冷接点感知装置の安定性に左右されます。最大限の精度と安定を維持するため、EX1048 は、入力端子と冷接点感知装置の接続の蓄熱量を大きく設計しています。これによって熱障害による影響を少なくできますが、動作環境によっては過渡的な測定誤差が生じることがあります。この様な環境を知っておくと、EX1048 の能力を最大限に引き出すことができます。

段階的な温度変化

EX1048 が、短期間に大きな環境温度の変化に晒された場合、EX1048 入力端子とその型特有の熱電対接点の間の蓄熱材と冷接点の内在している熱時定数の差により、過渡的な測定誤差が生じます。誤差の大きさは温度変化率に直接比例します。冷接点は、内蔵の蓄熱材に比べて外部温度に直接関わる為、外部温度変化に比較的良く追従します。そのため、環境温度が急激に上がった場合、冷接点の温度（冷接点測定も）は、蓄熱材より僅かに速いペースで温度が上昇します。結果として、チャンネル測定はマイナスの過渡誤差が生じます。逆に、環境温度が急激に下がった場合、チャンネル測定はプラスの過渡誤差が生じます。どちらの場合においても、温度の変化率が下がれば、熱安定は回復し、誤差がなくなります。この誤差は、校正によって修正できる物ではなく、熱安定の時間を設けることによって避けることができます。

ひとくちメモ　急な環境変化の後には、熱安定の時間を待ってから測定を開始してください。

端子の接続

似たような誤差は、EX1048 の入力チャンネルに雄型熱電対端子が最初に接続された時にも起こります。この場合の過渡誤差は、雄型熱電対端子を EX1048 本体の雌型端子に接続した時の温度差によって生じます。EX1048 本体と雄型端子が両方同じ温度環境にある場合は、この誤差は大体 3 分以内に 0.1 ºC 未満まで下がります。逆に、室温の端子が、温度の大きく異なる温度環境にある機器に接続された場合は、はるかに大きな誤差が生じ、無視できるレベルに下がるまである程度の時間が掛かります。この誤差は、校正によって修正できる物ではなく、端子接続後、熱安定の時間を設けることによって避けることができます。

ひとくちメモ　端子接続の後には、熱安定の時間を待ってから測定を開始してください。

隣接するチャンネルへの端子の接続

もうひとつの熱障害の源は、隣接するチャンネルへの端子の接続です。各冷接点補償センサは 4 つの入力チャンネルで共有される為、センサが監視しているどの 1 つのチャンネルに端子が接続されても、熱障害が観測されます。この障害は、新しく接続された端子が内部蓄熱材と違う温度の蓄熱材として働くからです。これによって、測定される冷接点補償（CJC）温度は微量ながら瞬時に下がります。同じCJC温度は他の 3 つのチャンネルでも使われる為、結果的に小さなマイナスの過渡誤差が生じます。この誤差の大きさは、一度に接続される端子の数に比例し、3 つ余計に接続された時に誤差が最大になります。この誤差は時間が経ち、全体が熱安定するとなくなります。一般的に最悪の場合でも、誤差は 3 分以内に 0.1 ºC 未満まで下がります。この誤差は、CJCセンサを共有しているチャンネル間でのみ生じます。また、端子の接続を断った時には、熱障害が生じないので、誤差もありません。この誤差は、校正によって修正できる物ではなく、端子群接続後、熱安定の時間を設けることによって避けることができます。CJCチャンネルと入力チャンネルの関連性については 表 1-4をご参照下さい。

ひとくちメモ　端子群接続の後には、熱安定の時間を設けてから測定を開始してください。

CJC #	チャンネル	入力チャンネル
CJC0	48	0-3
CJC1	49	4-7
CJC2	50	8-11
CJC3	51	12-15
CJC4	52	16-19
CJC5	53	20-23
CJC6	54	24-27
CJC7	55	28-31
CJC8	56	32-35
CJC9	57	36-39
CJC10	58	40-43
CJC11	59	44-47

表1-4: CJC チャンネル・入力チャンネルの関係

ハードウェアフィルタの選択

センサの帯域が必要でない限り、ハードウェアフィルタは外部ノイズの影響に強い 4 Hz に設定してください。

サンプリングレートの選択

サンプリングレートはデータ収集に必要な最低限の値に設定すると最高のノイズ性能を得られます。詳細は Section 3の *サンプリングレート・ノイズ性能*をご参照ください。

設置場所の選択

EX1048 は高・低温、強い気流、磁気などの源から離れた場所に設置してください。詳細は Section 2の*設置場所*をご参照ください。

熱電対線の選択

測定において、最良の結果を出す為には、できる限り短く、ゲージの太い熱電対をご利用ください。また、被覆熱電対を利用することによってシステムの電気干渉排除能力を高める事ができます。詳細は Section 2の *入力接点・配線*をご参照ください。

第 2 章

準備

製品の構成

EX1048 の梱包パッケージの中には、本体及び以下の部品が含まれております。

- EX1048 高精度熱電対計測器
- 電源ケーブル
- *EX1048 取扱説明書* (このマニュアル)
- *VTI Instruments LXI 取扱説明書*、*ドライバ*CD

ユニットを受け取り次第、損傷の無い事をご確認ください。

設置場所

本製品は基本的に電気、磁気、温度の干渉を受けないよう設計されていますが、以下の点を考慮することによって機器の性能をフルに引き出せます。

1) 本製品、特にフロント（前面）パネル部を高、低温ソースから離して設置してください。熱源となり得るほかの機器と一緒にラックマウントされる場合は、できるだけ離れた位置に設置してください。最低一台分のスペースを空ける必要があります。複数の EX1048 を利用する場合は直接重ねても性能に影響することなく使うことができます。
2) 本製品のフロントパネル部を強い空気流（通気口、機器やキャビネットファンなど）にさらさないでください。
3) 本製品をモーター、発電機、変圧器など強い磁場を発生するものの近くに設置しないでください。

ウォーミングアップ

EX1048 のウォームアップ時間は 60 分の仕様です。但し、計器が 5 ºC 以上の環境変動を経験する場合はハードウェアが安定するまで余計に時間をみることによって計器の性能を完全に引き出せます。

ネットワーク設定

デフォルトでは EX1048 は DHCP サーバ経由で IP アドレスを取得するよう設定されています。DHCP サーバが発見できない場合は、20 秒のタイムアウト後、AutoIP を利用して IP アドレス取得をします。

AutoIP は 169.254.X.Y の範囲で未使用の IP アドレスを割り当てる機能です。ここでは X は 1 – 254、Y は 0 – 255 の範囲を示します。機器はまず自分の MAC アドレスを元に最後から二番目のオクテットを X、最後のオクッテットを Y とした上で、169.254.X.Y のアドレスを割り当てます。ここで X に相当するオクテットが 0 の場合は 1 が、Y に相当するオクテットが 255 の場合は 254 が代わりに割り当てられます。このアドレスが既に使用されている場合は、割り当てのされていないアドレスを擬似乱数的に探し出します。

このAutoIP機能を例証したのが、表 2-1です。ここでは、各MACアドレスに対応した初期割り当てアドレスを表示しています。

MAC アドレス	AutoIP 初期アドレス
00:0D:3F:01:00:01	169.254.1.1
00:0D:3F:01:01:01	169.254.1.1
00:0D:3F:01:A3:28	169.254.163.40
00:0D:3F:01:FE:FE	169.254.254.254
00:0D:3F:01:FF:FE	169.254.254.254

表2-1: AUTOIP 初期割り当てアドレス

固定IPアドレスを割り当てる場合は、内蔵ウェブインターフェイスを使います。手順は、まず**Network Configuration**リンクをクリックし、DHCP割り当てを無効にしてください。この後、同画面で固定IPアドレスを割り当てます。詳細は Section 5の *Network Configuration*（ネットワーク設定）をご参照ください。

しかし、固定 IP アドレスを最も便利に利用する方法は、機器のほうでは DHCP を有効にしておき、会社の DHCP サーバ内で機器の IP アドレスを予約しておくという方法です。この方法では、予約 IP アドレスは DHCP サーバ内で機器の MAC アドレスと関連付けされ、EX1048 で設定しなくとも機器の起動と共に自動で割り当てられます。DHCP サーバで集中型 IP 配置をすることによって、管理を一箇所にまとめるだけでなく、複数の機器に同じ IP を割り当てる危険も回避できます。サーバの設定は会社の IT 事業部に相談してください。

EX1048 では VXI-11 のデバイス検出にも対応しています。これを使えば MAC アドレス、IP アドレスの予備情報がなくともブロードキャストを利用してローカルネットワーク内すべての EX1048 を発見できます。

リセットボタン

ネットワーク設定を間違った場合、またはネットワーク設定が不明な場合、EX1048 のリア（背面）パネル部にあるリセットボタンを使ってネットワーク設定を初期設定値に戻すことができます。ネットワークのリセットの手順は以下のとおりです。

1) EX1048 の電源を切ります。
2) リセットボタンを押しこみます。
3) ボタンを押したまま、電源を入れます。
4) そのまま 30 秒、リセットボタンを押したままでいます。
5) リセットボタンを放します。

EX1048 は直前のネットワーク設定でなく初期設定の値（DHCP）を使いますが、それ以外は普通に起動します。

時間設定

EX1048 は出荷状態ではSNTP (Simple Network Time Protocol) より時間を受け取るよう設定されています。また、ユーザーが手動で時間を設定することもできます。これは機器がインターネットにつなげられないようなネットワーク環境にある場合に必要になります。詳細は Section 5の *Time Configuration*（時間設定）をご参照ください。

入力接点・配線

EX1048 は、入力端子に無補償（Cu-Cu）mini-TC 雌型端子を利用します。この端子により、高い信頼性を維持できると共に、取り外しも簡単になります。また、熱電対の種類に依存しないので、ハードウェアの変更なしで複数の異種熱電対を 1 台で扱うことができます。入力端子は極性を持っている為、一方向にしか接続できません。接続端子は、標準的な mini-TC 雄型端子を利用します。一般的によく使われているのは、Omega Engineering 社の SMPW シリーズです。仕様精度を上げる為には、接続される熱電対線と同種の入力端子を利用する必要があります。

熱電対線は、極性を持っているので、熱電対線を熱電対端子に接続する時には、これに気をつける必要があります。以下の 表 2-2は、標準的な熱電対の色識別コードと極性をANSI（米国）、EIC（欧州）の両標準で示したものです。ご参照ください。

ANSI 熱電対規格				
熱電対	+		-	
J 熱電対	白		赤	
K 熱電対	黄		赤	
T 熱電対	青		赤	
E 熱電対	紫		赤	
S 熱電対	黒		赤	
R 熱電対	黒		赤	
B 熱電対	灰色		赤	
N 熱電対	オレンジ		赤	

IEC 熱電対規格				
熱電対	+		-	
J 熱電対	黒		白	
K 熱電対	緑		白	
T 熱電対	茶色		白	
E 熱電対	紫		白	
S 熱電対	オレンジ		白	
R 熱電対	オレンジ		白	
B 熱電対	灰色		白	
N 熱電対	ピンク		白	

表2-2: 標準熱電対参照図

ほとんどのアプリケーションにおいては、熱電対線の長さとゲージは測定精度には影響ありません。EX1048 のシグナルコンディション回路内で動的スイッチングが無い事と、高い入力インピーダンスにより、熱電対線の抵抗と静電容量は、通常、大きな要素とはなりません。しかし、システムの最大精度が求められる場合は、熱電対線の抵抗も誤差の計算に入れる必要があります。例として、200 m の 24 ゲージ T 熱電対は、約 500 Ω の抵抗値を持っています。EX1048 の一般的な入力バイアス電流 7.5 nA に対して、これによる電圧誤差は、

$$500\,\Omega \times 7.5\,\text{nA} = 3.75\,\mu\text{V}$$

となります。

この誤差を、対応する温度誤差に変換する為、今度は必要な熱電対特性曲線の角度で割ります。この場合、T 熱電対の 0 ºC での特性曲線の傾斜は、39 µV/ºC です。故に、ここでの温度誤差は、

$$3.75\,\mu V \div 39\,\mu V/°C = 0.1\,°C$$

となります。

この例は、特定の配線時に予測される誤差の評価方法を示します。ユーザーは、各測定において、誤差が満足できる範囲であることを確認することをお勧めします。一般的に、測定において最良の結果を出す為には、できる限り短く、ゲージの太い熱電対をご利用ください。

EX1048 は、その高いコモンモード除去性能と選択可能帯域制限によって、優秀なノイズ性能を提供し、理想的とはいえないような配線環境でも整合性の高いノイズフリーな測定を可能にします。それでも、幾つかの基本的な配線注意事項を守っていくことによって、最高レベルの性能を達成、保証することができます。

測定において、ユーザーは、被覆熱電対を利用することによってシステムの電気干渉排除能力を高める事ができます。被覆熱電対は、二つの熱電対線を、ユーザーがアースに繋ぐべく低インピーダンス導体で包みます。EX1048 は、この為に外部アースねじを提供しますが、アースであれば何でも構いません。

高電流の流れる時に必ず生じる磁気干渉は、二つの熱電対線によるループ面積を小さくする事によって、少なくできます。つまり、熱電対接合部から熱電対端子までの間、ワイヤーをできるくらい近くに並べてください。殆どの熱電対線は、二つの線を一緒にシールドで覆う事によって、始めからループ面積を最小化しています。

多くのテストアプリケーションでは、測定物をチェンバーの中で監視しますので、多種の信号をチェンバーのケーブルアクセスポートに通す必要があります。この時、熱電対線は、電力や高周波信号のケーブルとは違うポートをご利用することをお勧めします。

第 3 章

基本操作

はじめに

この章では EX1048 の機能と利用方法をもう少し詳しく説明します。

工学単位(EU)変換

本製品の入力チャンネルは 11 種類の工学単位変換を、チャンネル毎に 1 種類ずつ設定できます。それぞれの種類と定義は下の表をご参照ください。

単位変換	パラメータ
電圧	MV
J 熱電対	J
K 熱電対	K
T 熱電対	T
E 熱電対	E
S 熱電対	S
R 熱電対	R
B 熱電対	B
N 熱電対	N
ユーザー定義 0	User0
ユーザー定義 1	User1

表3-1: 工学単位変換設定

電圧 入力で測定された元の電圧を、ボルト（V）の単位で返します。チャンネルの測定温度、入力冷接点補償温度には影響されません。

J, K, T, E, S, R, B, N 熱電対 入力で測定された熱電対温度を補償して温度（ºC 又は ºF）の単位で返します。熱電対計算は NIST ITS-90 熱電対仕様表の全次数多項式、また係数を使います。

ユーザー定義 0, ユーザー定義 1 入力で測定された熱電対温度を補償して温度（ºC又はºF）の単位で返します。熱電対計算はユーザーが定義した係数を使います。詳細は *ユーザー定義変換式*をご参照ください。

初期設定は「電圧」です。

ひとくちメモ 熱電対が数種類使われている環境では、熱電対と計器の設定を間違えて組み合わせ易くなります。そういった誤差は計器の精度よりかなり大きいですが、見てすぐ分かる程大きくない場合が多いので、気をつけてください。

ハードウェアフィルタ

EX1048 の入力はチャンネル毎 にハードウェアフィルタを 4 Hz、または 1 kHz の遮断周波数に設定できます。 この二つのうち、ほとんどの温度測定用途では、中ゲージ熱電対信号を歪曲することなく十分なレスポンスを提供し、高レベルの(50/60) Hz コモンモード除去を誇る 4 Hz が適切です。また、この設定は最低レベルのノイズフロアを提供する為、高周波数を必要としない全ての用途でお勧めします。高周波数が必要である場合は、1 kHz 設定が使えます。そのレスポンスは最も高速な細ゲージ熱電対の出力でもほとんど歪み無くパスします。

初期設定では 4 Hz が選択されています。

測定範囲・入力保護

EX1048 の仕様入力電圧範囲は±66 mV です。この値は無歪みでの最高差動電圧（＋入力端子とー入力端子の間の電圧）を表しています。差動測定に仕様外の歪みを作らない最高コモンモード電圧（＋とー入力両方に共通する電圧）は±10 V です。この範囲外の電圧入力は不正確な測定に繋がりますが、±35 V までなら機器が破損することはありません。この限界値を超える電圧を加えた場合は EX1048 の永久破損につながることがあります。

ひとくちメモ ±35 V 以上の電圧入力は本製品の永久破損につながることがあります。

EX1048 の温度測定範囲は入力電圧範囲と熱電対自体の能力に影響されます。一般的な熱電対に於ける EX1048 の測定範囲は以下の通りです。

	最低値(°C)	最高値(°C)	最低値(°F)	最高値(°F)
J 熱電対	-200	1200	-328	2192
K 熱電対	-200	1372	-328	2502
T 熱電対	-200	400	-328	752
E 熱電対	-200	900	-328	1652
S 熱電対	-50	1768	-58	3214
R 熱電対	-50	1768	-58	3214
B 熱電対	250	1820	482	3308
N 熱電対	-200	1300	-328	2372

表3-2: EX1048 測定範囲

これらの標準的な熱電対のほかにも EX1048 ではカスタム設定によりどんなタイプの熱電対にも対応できます。この場合、温度測定範囲は EX1048 の入力電圧範囲とカスタム熱電対の変換式より計算されます。

冷接点補償(CJC)

最も高レベルな精度と安定性を提供するために EX1048 は熱電対 4 チャンネルにつき 1 つ、計 12 のサーミスタによって監視された恒温入力部を備えています。CJC データが常に入力チャンネルと時間的に相互関連していることを保証するため、CJC チャンネルはスキャン毎に測定され、入力チャンネルの測定と対応する CJC チャンネルの測定の時間的ずれは 4 ms 以内に制限されています。CJC チャンネルと入力チャンネルの関係は以下の通りです。

CJC #	チャンネル	入力チャンネル
CJC0	48	0-3
CJC1	49	4-7
CJC2	50	8-11
CJC3	51	12-15
CJC4	52	16-19
CJC5	53	20-23
CJC6	54	24-27
CJC7	55	28-31
CJC8	56	32-35
CJC9	57	36-39
CJC10	58	40-43
CJC11	59	44-47

表3-3: CJC チャンネル・入力チャンネルの関係

測定された CJC データの報告の有無はユーザー側によって設定可能です。これはデータの表示の管理であり、実際の測定には影響ありません。報告設定にかかわらず CJC データはスキャン毎に更新されます。また、報告が有効になっている場合は入力チャンネルの°C/°F 設定に関係なく ºC の値で報告されます。

EX1048 は、また、ユーザーによって管理・測定された外部冷接点の使用にも対応しています。この場合、冷接点温度は ºC で入力され、チャンネル毎に設定の無効・有効を決めます。つまり、内蔵・ユーザー定義の CJC 入力は併用可能だということです。また、この設定は入力チャンネルの°C/°F 設定には影響されません。

初期設定で選択されているのは、

- CJC 報告は無効
- ユーザー定義 CJC 温度は 0.0°C
- ユーザー定義 CJC 温度は全チャンネルにおいて無効

温度単位

EX1048 は温度データを ºC、又は ºF の 単位で出力できます。この設定は全体的に適用されますので、一つの設定で全入力チャンネルに対して影響します。この選択は入力チャンネルのデータにしか適用されません。冷接点測定データや外部冷接点温度の入力値は ºC に固定されています。同じく、EU 電圧変換に設定されている入力にも影響はありません。

初期設定値は「°C」です。

サンプリングレート・ノイズ性能

EX1048 は、スキャンリスト上のチャンネル数に関わらず、最高 1 kHz のサンプリングレートを設定できます。選択されたサンプリングレートは、スキャンリスト全体が測定される周波数を指します。そのため、全チャンネルは同じサンプリングレートで測定されます。但し、サンプリングレートが 1 kHz より著しく少ない場合は EX1048 が自動的に複数のサンプルを取り、その平均値を算出します。こうすることによって、EX1048 は要求された速度でデータ出力をしながらノイズ性能を上げることができます。以下に表示されているのは、選択されたサンプリングレートにおいて、25 ºC の環境で K 熱電対を 100 回ずつ読み取った場合のノイズプロファイルです。

サンプリングレート	ノイズ (ºCp-p)
1 kHz	0.26
500 Hz	0.13
400 Hz	0.10
300 Hz	0.08
≤ 200 Hz	0.07

表3-4: K 熱電対ノイズプロファイル対サンプリングレート

上の表を見ると分かるように、サンプリングレートが 200 Hz に達すると平均されるサンプルの数は最高になり、機器のノイズフロアを示します。これ以降サンプリングレートを落としても、ノイズプロファイルは下がりません。実際には、EX1048 のノイズ性能は利用されるセンサの熱電対特性の曲線角度の関数として見ることができます。つまり、E 熱電対 (59 µV/ºC) は性質上 K 熱電対(39 µV/ºC)よりきれいな測定を提供します。しかし、上記のノイズとサンプリングレートの関係は、どの種類の熱電対にも当てはまります。そのため、最高のノイズ性能を保証するためには、サンプリングレートをデータ収集の必要としている最低値に設定することをお勧めします。

ひとくちメモ ここで説明された平均化処理は、トリガモデルがタイマーのTRIGソースに設定されているときのみ動作します。詳細は Section 4の *測定性能の最適化*をご参照ください。

スキャンリスト設定

EX1048 は、1 から 48 までの入力チャンネルをスキャンリストに設定できます。設計のチャンネル独立性の為、スキャンリストの構成が精度、ノイズ、速度へ影響することはありません。チャンネル入力は、純粋にユーザーのアプリケーション要件によることができます。正しく設定されたスキャンリストは、以下の条件を満たしている必要があります。

- 最低 1 チャンネルを含む。
- 48 チャンネル以下。
- チャンネルの重複がない。

内蔵ウェブインターフェイスで入力されたスキャンリストは、数字の小さいチャンネルから順にスキャンされます。スキャンの順序にもう少し柔軟性が求められる場合は、機器のドライバ経由で好きな順番のスキャンが設定できます。

EX1048 の各入力チャンネルは、スキャンリスト上の有無に関わらず高い入力インピーダンスと他チャンネルから独立した動作を維持します。つまり、スキャンされないチャンネルの入力信号との接続の有無は動作に影響ありません。

スキャンリストタイミング

スキャンリスト上のチャンネル数、外部CJC温度の使用、入力チャンネルの工学単位変換係数に関わらず、各スキャンシーケンスは、12 のCJCチャンネルを順番に測定開始します。この後、選択されたスキャンリストチャンネルは個別に測定されます。測定チャンネル間の時間差を最小に留める為、スキャンシーケンサはスキャンリストを可能な限り高速に循環します。*サンプリングレート・ノイズ性能*の項で取り上げられた平均化処理の為、チャンネル間の正確なタイミングは選択されたサンプリングレートによります。以下の表 3-5 では、選択されたサンプリングレートにおけるチャンネル間タイミングを表記しています。

サンプリングレート	チャンネル間タイミング (µs)
1 kHz	16.6
500 Hz	32.1
400 Hz	39.8
300 Hz	51.5
≤ 200 Hz	74.7

表3-5: チャンネル間タイミング対サンプリングレート

過渡信号の時間差を最小限に留める必要のあるアプリケーションに於いては、以下の設定変更を行うと、機器を最高速に構成し直します。

- サンプリングレートを 1 kHz に設定する。
- スキャンリストから、不要なチャンネルを削除する。
- ハードウェアフィルタの 1 kHz 設定を使用する。

自己校正

広い動作環境範囲に於いて、高い測定精度を実現する為、EX1048 は、自己校正の機能を備えています。自己校正時には、入力シグナルコンディションパスは入力端子の接続を断ち、内蔵校正ソースで駆動する校正バスに接続されます。複数の校正ソースポイントに於いてコンディションパスを測定することによって、最後の完全校正時からの回路のドリフトに対してソフトウェアで補償が行われます。一度自己校正されると、表 1-2 の *熱電対精度（自己校正使用時）* で記されている数値は±2 °C までの仕様環境変化に於いて、30 日間まで有効です。自己校正の使用に関わらず、EX1048 は、最低一年に一度、完全校正をする必要があります。

自己校正は、可能な限り、なるべく頻繁に行ってください。特に、前の校正時から環境温度が大きく変わった場合は、自己校正の使用は大事になります。ただ、急な環境変化の後には、自己校正の使用は、一定の熱安定を待ってから行うことが望ましいです。自己校正作業は、短時間で終わる上、ユーザーが入力端子を切断したり外部機器を利用する必要がないので、簡単に何度でも行えます。

また、自己校正は、EX1048 の電源を入れてから最低 60 分のウォームアップ時間を経多後に、行ってください。ユーザー保護の為、ウォームアップ時間終了前に自己校正が実行された場合は、警告メッセージが表示されます。しかし、これは単なる警告であり、自己校正命令を再発行することによって無視できます。警告を無視しても問題の無いケースは、a) 機器が既にウォームアップしており、直に違う場所に移動された場合、又は b) 施設内が停電になり、一時的に機器の電源が切れた場合です。

自己校正によって、完全校正時に生成された不揮発性メモリ内の校正定数が上書き、変更、置き換えられることはありません。代わりに、測定計算に完全校正定数が適用された後、これを補足する付加的な校正定数を生成します。初期設定では、自己校正データは揮発性メモリ内に置かれ、機器のリセット、又は電源の再投入と共に消去されます。こうする事によって、機器は必ず完全校正時の校正定数が入った状態で起動します。これによって、ユーザーは他ユーザーの使用環境に影響されなくなる為、この機能は機器が複数のユーザーで共有されるときに重要になります。

いつでも行えるこの自己校正ですが、自己校正が必要でありながら、不揮発性のデータが要求されるケースもあります。EX1048 はそのような利用にも対応しています。いったん機器が自己校正されれば、別命令により不揮発性メモリ内に保存できます。また、以前に保存された自己校正データは、不揮発性メモリよりロード、及び消去できます。

自己校正は、EX1048 のシグナルコンディション回路に対する時間と温度の影響を軽減し、大幅に測定性能を高めます。しかし、極端な外部温度源や複数の接点を同時接続後に熱安定を待たないで測定した場合の冷接点熱安定誤差の補償はできません。また、校正ループ外の極端に長い熱電対線の電圧減による誤差も軽減されません。詳細は、Section 1の *測定性能の最適化*を参照ください。

自己校正はEX1048 のドライバ、及び内蔵ウェブ画面のインターフェイスにより操作できます。ウェブ画面操作に関しては Section 5の *Self Calibration（自己校正）*をご参照ください。プログラミング操作に関しては Section 6の *自己校正の実行*をご参照ください。

熱電対断線検出・限界値

EX1048 は、熱電対断線検出および汎用入力チャンネルモニタリングに使われる二対のプログラマブル限界値を提供します。これらの限界値、リミットセット 0 とリミットセット 1 は、チャンネル毎にプログラム可能です。限界値評価は、各測定完了時に行われ、最大 25 ms のレイテンシで更新されます。

各リミットセットは、上限値と下限値を持っており、限界値評価はいずれか、または両方に対して行うことができます。限界値評価の出力は、3 つの形で表示されます。フロントパネル LED の動作は、リミットセット 0 の上限値と下限値に結び付けられています。デジタル I/O ポートの動作は、ユーザーの上限値、下限値の組み合わせをどちらか、または両方の限界値セットに関連付けが可能です。最後に、限界値状態の情報は、機器のドライバより確認が可能です。

リミットセット 0 は通常、熱電対断線検出に使われるフロントパネルLEDに結び付けられている為、リミットセット 0 にはリミットセット 1 には無い、特殊な動作機能を持っています。リミットセット 0 の値は、初期設定で、各チャンネルの工学単位変換タイプと温度単位設定を基に自動的に決められます。つまり、リミットセット 0 の上限値と下限値は、表 3-2で表記されているEX1048 の測定範囲の上限値と下限値に設定されるのです。もし必要とされるなら、この自動設定を無効にし、手動で上限値と下限値を入力することもできます。この手動動作モードでは、限界値は工学単位変換タイプ及び温度単位設定の変更に関わらず同じ値を保ちます。リミットセット 1 は、手動モードでのみ働きます。

前述したように、限界値評価はデジタルI/Oポートの動作に関連付けることもできます。詳細は、*デジタルI/O*とDIOリミットイベントの項をご参照ください。

EX1048 は、リア（背面）パネル部に 8 チャンネルデジタル I/O ポートを備えています。このポートは、アーム／トリガソース、限界値評価情報の表示、及び汎用出力デバイス

として利用可能です。汎用出力デバイスとして使う場合、各 DIO（デジタル I/O）チャンネル は、出力時の出力機能と取るべきスタティックレベルを個別にプログラムすることができます。作業をより拡張、自動化するため、各 DIO チャンネルは独立して 1 ないし複数の入力チャンネルに対して、1 ないし複数の限界値状況に関連付けできます。

限界値状態の初期化は、各表示メカニズムによって少しずつ違ってきます。フロントパネル LED の場合は、常にラッチ設定で動作、つまり収集シーケンス内で後に続くチャンネル測定値が限界値内でもチャンネルの LED は点灯し続けます。これは、接続状況が不安定な熱電対の検出に重要な機能です。限界値状態は新しい収集シーケンスの初めにクリアされます。DIO リミットイベントは、ラッチ、非ラッチモードのどちらにもプログラムできます。こちらも新しい収集シーケンスの初めにクリアされます。最後に、累積した限界値状態の情報は機器のドライバより入手可能で、新しい収集シーケンスの初めにやはりクリアされます。

初期設定では、各限界値は最高値になっています。

デジタル I/O と DIO リミットイベント

EX1048 は、リア（背面）パネル部に 8 チャンネルデジタル I/O ポートを備えています。このポートは、アーム・トリガソース、限界値評価情報の表示、及び汎用出力デバイスとして利用可能です。デジタル I/O の接続は以下のピン配列をした標準の DB-9 コネクタを使います。

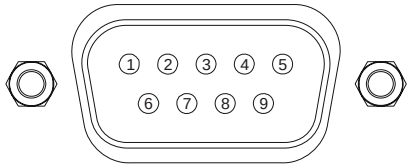


図3-1: デジタル I/O DB-9 コネクタ

ピン	機能
1	DIO Channel 0
2	DIO Channel 1
3	DIO Channel 2
4	DIO Channel 3
5	DIO Channel 4
6	DIO Channel 5
7	DIO Channel 6
8	DIO Channel 7
9	GND

表3-6: デジタル I/O コネクタピン配列

汎用出力デバイスとして使う場合、各DIO（デジタルI/O）チャンネル は、出力時の出力機能と取るべきスタティックレベルを個別にプログラムすることができます。もし出

力として適用されていない場合は、他の電圧ドライバとの競合を防ぐ為、チャンネルは 3 ステート状態になります。ポートの電圧許容限界と出力駆動能力は 表 3-7のポート仕様をご参照ください。各チャンネルは、出力設定に関係なく、入力を今まで通りに受け付けます。つまり、各チャンネルの入力レベルは、特別な適用命令なしでアクセスできます。また、DIOチャンネルの基本機能は、トリガ、スキャン、その他の機器処理に影響されることはありません。この後説明する限界値状態に関連付けされない限り、DIO チャンネルの動作は完全に自動で行われます。

また、出力として適用されている場合、各チャンネルは命令とともに 1 µs パルスを生成することができます。このパルスの使用例としては、EX1048 での他の測定器のトリガがあります。このパルス命令は、その特定のチャンネルにプログラムされたスタティックレベルに依存します。チャンネルのスタティックレベルが high にプログラムされている場合、パルスは low になります。チャンネルのスタティックレベルが low にプログラムされている場合は、パルスは high になります。各パルス生成は、個別に命令を実行する必要があります。

自動化するため、各 DIO チャンネルは個別に 1 ないし複数の入力チャンネルに対して、1 ないし複数の限界値状態に関連付けできます。これは DIO リミットイベントと呼ばれます。例えば、DIO チャンネル 0 は、チャンネル 2、セット 0 の上限値かチャンネル 1、セット 1 の下限値を超えたときに high になるようプログラムできます。リミットイベントとして関連付けされた場合、DIO チャンネルは新しい収集シーケンスの初めにクリアされます。この後、DIO ステートはプログラムされた限界値評価条件にそって各スキャン毎に更新されます。初期設定では、DIO ステートはクリア時に low になりますが、Invert（反転）設定によりチャンネル単位で high にセットすることもできます。同じく、各チャンネルの初期動作は非ラッチモードですが、チャンネル単位でラッチモードにセットできます。ラッチモードでは、一度 DIO リミット値に達すると、次の収集シーケンスの初めにクリアされるまで値を保ちます。

この機能を利用するために、ひとつ知っておくと良いことは、DIO リミットイベント割り当てによる DIO チャンネルの操作は、直接出力メカニズムのロックアウトを行わないということです。例えば、DIO チャンネルが DIO リミットイベントによって high になっていても、直接出力操作によって非同期的に low に変えることができます。限界値の評価はデータ収集速度を超えることはできないので、DIO の直接操作は、低速サンプリングレートの使用時に、普通の限界値評価処理より早く DIO チャンネルのリセットを行いたいという場合に、これを可能にします。このメカニズムは、DIO チャンネル出力値を非同期に変更するもので、リミットイベントメカニズムを変更したり、無効にしたりすることはありません。次のスキャン（そしてそれ以降の限界値評価）からは、DIO リミットイベントの設定通り、普通に DIO チャンネルの更新が行われます。一般的には、アプリケーションはひとつの操作メカニズムしか利用しません。その為、誤使用を防ぐ為に、DIO チャンネルが DIO リミットイベントと関連付けされている場合は、その旨表示されます。

各 DIO チャンネルの初期選択値は、

- 出力適用はオフ
- 出力レベルは 0

DIO 限界値イベントの初期選択値は、

- 全チャンネル無効
- 反転はオフ
- ラッチはオフ

表 3-7は、デジタルI/Oポートの電気仕様を表記しています。特に注意して頂きたいのは、-0.5 V ~ 5.5 VのV_{INPUT} の仕様です。抵抗がある場合でも、外部電源電圧がこの値を超えるとEX1048 の破損に繋がる場合があります。

特性	値
V_{INPUT}	-0.5 V ~ 5.5 V
V_{IH}	2 V min
V_{IL}	0.8 V max
V_{OH} (I_{OH} = -32 mA)	2 V min
V_{OL} (I_{OL} = 64 mA)	0.55 V max

表3-7: デジタル I/O ポート電気仕様

LXI トリガバス

EX1048 は、機器リア（背面）パネル部に 8 チャンネル LXI トリガバス（VTB）を備えています。この差動ペア LVDS バスは、並列接続された二つの同種ポートによって構成されています。トリガバスの主用途はマルチユニットトリガ、及び同期用の高速信号伝達ですが、汎用出力デバイスとしても利用できます。この場合、他の LVDS ポートとの接続を想定しており、汎用デジタル I/O ポートとの接続はできません。トリガバスコネクタは、以下のピン配列に沿ったマイクロ DB-25 コネクタです。

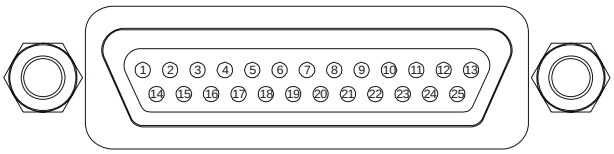


図3-2: トリガバス DB-25 コネクタ

ピン	機能	ピン	機能
1	GND	14	VTB Channel 0 正
2	VTB Channel 0 負	15	GND
3	VTB Channel 1 正	16	VTB Channel 1 負
4	GND	17	VTB Channel 2 正
5	VTB Channel 2 負	18	GND
6	VTB Channel 3 正	19	VTB Channel 3 負
7	GND	20	VTB Channel 4 正
8	VTB Channel 4 負	21	GND
9	VTB Channel 5 正	22	VTB Channel 5 負
10	GND	23	VTB Channel 6 正
11	VTB Channel 6 負	24	GND
12	VTB Channel 7 正	25	VTB Channel 7 負
13	GND		

表3-8: トリガバスコネクタピン配列 [1]

トリガバスコネクタとの接続はVTI Instruments トリガバスケーブル (P/N: 70-0307-001)を使用します。正常な動作の為には、各トリガバスコネクタはケーブル、又はターミネータと繋がっている必要があります。各ユニットが二つ備えたこのコネクタは、並列しているので、ケーブル、ターミネータの接続はどちらを選んでも同等となります。機器の

[1] このピン配列は EX1048 rev. A バージョンのものです。Rev. B バージョンのピン配列をお求めの場合は *www.vtiinstruments.com* で最新のマニュアルをダウンロードしてください。

台数に関わらず、ターミネータは両端に一つずつ、計二つしか必要でありません。VTI Instruments P/N: 70-0304-000 はこの二つのターミネータを提供します。図 3-3は、3ユニットの場合のトリガバス取り付け例です。

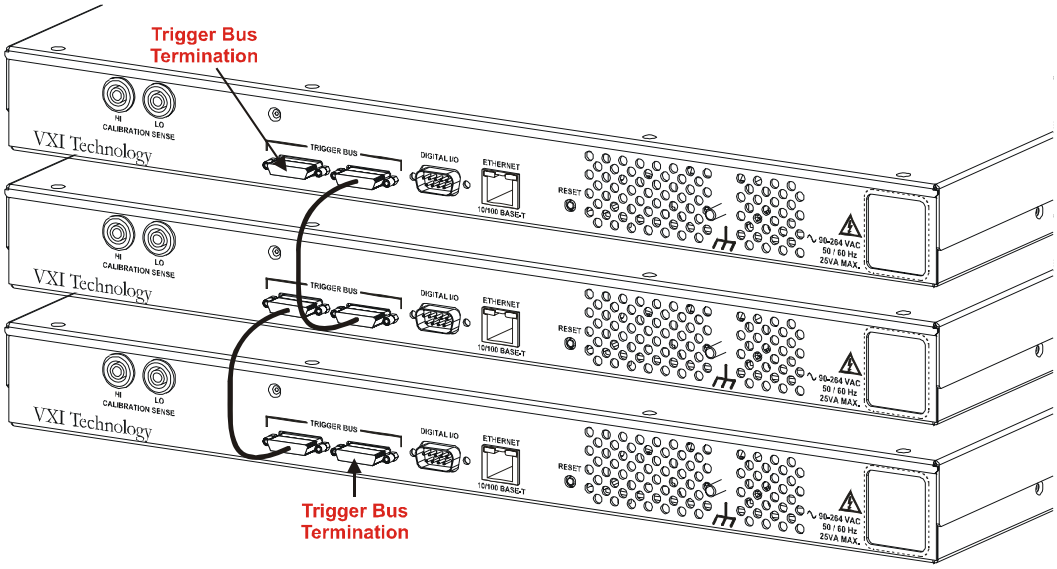


図3-3: トリガバス配線例

機器を汎用出力デバイスとして使う場合、各 VTB チャンネルの出力機能、及びスタティックレベルを個別にプログラムすることができます。もし出力として設定されていない場合は、他の電圧ドライバとの競合を防ぐ為、チャンネルは 3 ステート状態になります。出力機能に関わらず、各チャンネルは一定の入力機能を提供します。つまり、特別な適用命令なしで各チャンネルの入力レベルがアクセスできるのです。また、VTB チャンネルの基本機能は、トリガ、スキャン、その他の機器処理に影響されることはありません。

また、出力として設定されている場合、各チャンネルは命令によって 1 µs パルスを生成することができます。このパルスの極性は、その個々のチャンネルにプログラムされたスタティックレベルによって決まります。チャンネルのスタティックレベルが high にプログラムされている場合、パルスは low になります。チャンネルのスタティックレベルが low にプログラムされている場合は、パルスは high になります。各パルスを発生するには、個別に命令を実行する必要があります。

このパルス生成の一般的な利用方法としては、マルチユニットの同期があります。上記の 図 3-3を例に取ると、すべてのユニットを同じVTBチャンネルでトリガする様、設定できます。 1 台のユニットがマスターとして選択され、3 台の機器の共通トリガソースとして、指定されたVTBチャンネル線上に信号を出力します。

各 VTB チャンネルの初期選択値は、

- 出力適用はオフ
- 出力レベルは 0

トリガバスの電気仕様は、表 3-9をご覧ください。

特性	値
Logic Type	M-LVDS Type 2
V_{IT+}	150 mV max
V_{IT-}	50 mV min
V_{OS}	1 V typical

表3-9: トリガバス電気仕様

ロック

初期設定では、EX1048 はウェブ及び機器ドライバの両インターフェイスで、複数のホストによる無制限のアクセスを許可します。これはユーザーに高い柔軟性をもたらしますが、プロテクトされた作業が望ましい、または重要なケースもあります。そのような場合は、EX1048 を“ロック”することによってこのロック命令を出した IP ホスト以外からの命令を受け付けなくすることができます。

このロックメカニズムは、設計上、他ホストからの“ロック切断”命令によって無効にすることが可能です。この為、ロック機能は他ユーザーに対して、プロテクトされた作業が実行中であることの通知にはなりますが、安全の保障をする物ではありません。これにより、ロックを実行した IP アドレスが利用不能になった場合も、機器のリカバリーが可能になっています。

自己校正を実行する場合、事前に機器のロックを取得する必要があります。

トリガ

EX1048 では、独自のアームソースとトリガソースイベント構造を備えたフルファンクショントリガモデルをサポートしています。トリガモデルに関する詳しい説明は Section 4をご参照ください。要約すると、収集シーケンスは、トリガ開始命令によって開始されます。この後、プログラムされたアームソースイベントの受け取り、続くトリガソースイベントの受け取りと共に、スキャンが始まります。トリガ、及びアームソースイベントは、Immediate（即時）、Timer（タイマー）、Digital I/O（デジタルI/O）、Trigger Bus（トリガバス）など無数のソースから個別にプログラム可能です。

データ形式

初期設定では、データ読み出し時に得られる情報はチャンネル読み取り情報とスキャン開始の時刻に限られます。EX1048 では、この他に測定された CJC（冷接点補償）温度とチャンネル読み取り時刻（スキャン始めからのデルタ値として）を返すことができます。

データ収集

一般的に EX1048 でのデータ収集、及び読み出しは、時間的に間隔を置いた離散的な命令によって行われます。EX1048 は収集データを読み出し前にバッファする為、20 MB の FIFO メモリを利用します。このリーディングバッファは、データ書き込みの用意の為、トリガ開始命令の受け取りと共に消去されます。そして、プログラムされたトリガモデル条件が満たされ次第、EX1048 は、スキャンリストで定義されたチャンネル設定通りにスキャンを始めます。スキャンと収集データの保存は、トリガとアームカウント値に達するかスキャンが中止されるまで続きます。この時点でスキャンは停止し、トリガモデルは INIT レイヤーに戻ります。

バッファ可能なスキャン数はスキャンリスト上のチャンネル数とデータフォーマットによって決まります。この為、特に長いスキャンシーケンス（カウント値の意味で）においては、データフォーマットを小さくする事によって保存可能なスキャン数を増やすことができます。厳密には、バッファ可能なスキャン数（ページカウント）は以下の公式で求めることができます。

$$\text{Channel_Size} = 4 + 2 \times \text{Timestamp_Reporting}$$
$$\text{CJC_Channel_Size} = 4 + 2 \times \text{Timestamp_Reporting}$$
$$\text{CJC_Channel_Count} = 16 \times \text{CJC_Reporting}$$
$$\text{Page_Size} = 108 + \text{Channel_Count} \times \text{Channel_Size} + \text{CJC_Channel_Count} \times \text{CJC_Channel_Size}$$
$$\text{Page_Count} = 20971520 / \text{Page_Size}$$

ここでは、

Channel_Count	1-48（スキャンリスト上のチャンネル数）
CJC_Reporting	0 or 1 (1 = YES, 0 = NO)
Timestamp_Reporting	0 or 1 (1 = YES, 0 = NO)

一般的な構成における Page_Count（ページカウント）サイズは以下の通りです。

Channel_Count	CJC_Reporting	Timestamp_Reporting	Page_Count
1	0	0	187246
16	0	0	121927
32	0	0	88862
48	0	0	69905
48	1	0	57614
48	0	1	52958
48	1	1	42625

表3-10: ページカウントサイズ例

スキャン中にリーディングバッファがいっぱいになった場合、それ以降の測定データの取り扱いは、機器のブロッキングモードの設定で決まります。ブロッキングモードが適用されている場合は、それ以降のデータは放棄され、バッファの情報はそのまま保持されます。ブロッキングモードが適用されていない場合は、新しいデータはバッファの情報を古い順から上書きしていきます。

ブロッキングモードの初期設定値は「無効」です。

ひとくちメモ リーディングバッファは、揮発性メモリ上にあるので、機器のリセットや再起動において消去されます。

データ読み出し

通常、データは収集シーケンスの終わりに EX1048 から読み出されます。この動作方式は内蔵ウェブインターフェイス、及び機器のドライバにおいて完全にサポートされています。また、ドライバを使うと収集シーケンス中にデータを読み出すことが可能です。どちらの方式を選んでも読み出しデータは EX1048 によって校正、補償され、選択された単位で返ってきます。ユーザーによるデータ編集の必要はありません。データは、ページ（スキャン）毎、または複数ページのブロックとして読み出すことができます。将来のデータ収集時に最大限のリーディングバッファ容量を確保する為、FIFO メモリ内のデータは回収と共に消去されます。

ユーザー定義変換式

EX1048 は標準的な熱電対に対応し、NIST ITS-90 熱電対データベースの全次数多項式、または係数を使って熱電対計算をします。しかし、アプリケーションによっては、機器内の係数でなくユーザーの定義した係数を使いたい場合もあります。この理由としては幾つかありますが、その一つは、利用される熱電対の伝達関数が標準的な熱電対の誤差範囲より高い精度で得られた場合。もうひとつは、標準的でない熱電対を利用する場合です。係数は最高二組が入力可能です。すなわち、ユーザー定義熱電対式を利用するにはユーザーが二つの変換公式を定義する必要があります。

*forward conversion polynomial（前方変換式）*は、CJC 温度を補償冷接点電圧に変換するときに使われ、次のように記されます。

$$E = c_0 + c_1 * t^1 + c_2 * t^2 + \ldots + c_{12} * t^{12}$$

ここでは、*E* はボルト、*t* は ºC の単位、そして c_0 - c_{12} が係数です。

*inverse conversion polynomial（逆変換式）*は、補償入力電圧を温度に変換するときに使われ、次のように記されます。

$$t = d_0 + d_1 * E^1 + d_2 * E^2 + \ldots + d_{12} * E^{12}$$

ここでは、*E* はボルト、*t* は ºC の単位、そして d_0 - d_{12} が係数です。

係数の初期設定値は「0.0」です。

ひとくちメモ ユーザー定義の係数は、入力しただけでは測定に使用されません。使用するには工学単位変換を User0 か User1 に設定してください。

第 4 章

トリガ

はじめに

EX1048 では、独自のアームソースとトリガソースイベント構造を備えたフルファンクションントリガモデルをサポートしています。トリガモデルは業界標準のSCPI 1999 トリガサブシステムを基本にしており、図 4-1で示されています。

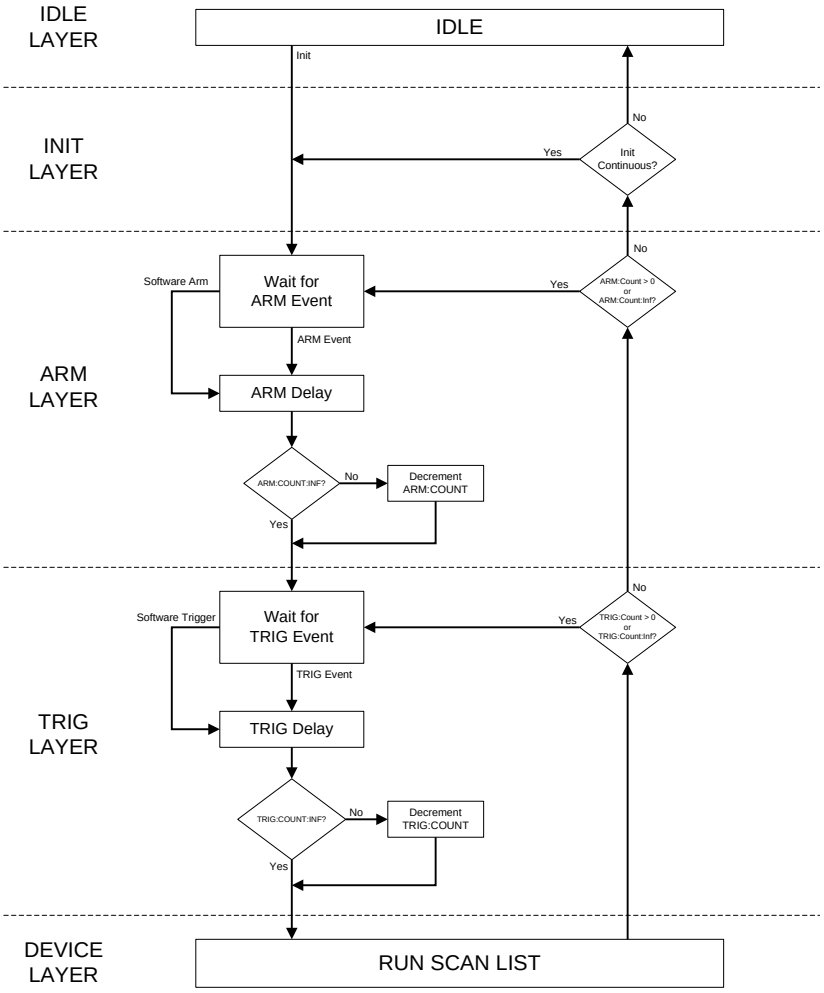


図4-1: EX1048 トリガモデル

トリガモデルは、5 つのレイヤーによって構成されます。これは、IDLE、INIT、ARM、TRIG、DEVICE です。EX1048 のリセットは、機器を IDLE 状態に置きます。トリガ開始命令は、機器を INIT レイヤーを経由して ARM レイヤーに移行することによって、収集シーケンスを始めます。この時、リーディングバッファメモリは消去され、適用された限界値イベントはすべて初期設定値に戻されます。

ARM レイヤーに入ると、ARM Count は指定された値に戻されます。この後、機器は指定された ARM イベントが発生するかソフトウェア・アームが出されるまで ARM レイヤーで待機します。条件が満たされると、指定された ARM Delay 時間が待たれ、ARM Count 値は減らされます。そして機器は TRIG レイヤーに移行します。

TRIG レイヤーに入ると、TRIG Count は指定された値に戻されます。この後、機器は指定された TRIG イベントが発生するかソフトウェア・トリガが出されるまで TRIG レイヤーで待機します。条件が満たされると、指定された TRIG Delay 時間が待たれ、TRIG Count 値は 1 つ減ります。そして機器は DEVICE レイヤーに移行します。

DEVICE レイヤーでは、スキャンリストが測定、校正され、ローカルメモリに保存されます。また、限界値に達しているか評価され、適用された限界値イベント状態が更新されます。

機器は TRIG カウントが 0 になるまで（つまり指定された数の TRIG イベント、そしてその後のデバイス動作が行われるまで）、TRIG レイヤーに留まります。いったん TRIG カウントが 0 に達すると、EX1048 は ARM カウントの残り値を評価し、0 でなければ ARM レイヤー動作を繰り返します。ただ、TRIG レイヤーへの移行時には毎回 TRIG カウント値をリセットする為、各 ARM レイヤー動作は TRIG レイヤーと DEVICE レイヤー上の全動作を繰り返すことになります。

いったん ARM カウント値が 0 に達すると、機器は INIT レイヤーに戻ります。Init Continuous モードが適用されている場合、トリガ開始命令を出さなくても ARM レイヤーは自動的に再入力されます。しかし、トリガの開始とは違い、リミットイベントはリセットされませんし、リーディングバッファも消去されません。逆に、Init Continuous モードが無効になっている場合、機器は IDLE レイヤーに戻り、新しい収集シーケンスを始めるにはトリガ開始命令を再発行する必要があります。

イベント

トリガモデル内の動作は ARM イベント、TRIG イベントのふたつのイベントにより制御されています。これらのイベントはトリガバス、デジタル I/O ポート、システムタイマーの好きな組み合わせによって発動するよう、個々にプログラムすることができます。また、各イベントは Immediate（即座）動作するようにプログラムすることによってイベントモニタ条件をを常に満たした状態にすることができます。各イベントモニターは、また、必要に応じて Software Arm（ソフトウェア・アーム）、Software Trigger（ソフトウェア・トリガ）を発行することによって回避することができます。ソフトウェア・アームとソフトウェア・トリガはプログラムされたイベントソースに関係なく常に適用されます。

Digital I/O Port（デジタル I/O ポート）と Trigger Bus（トリガバス）は機器のリア（背面）パネル部にあるデジタルハードウェアポートをモニタします。イベントは各ポートの 8 チャンネルの任意の組み合わせで制御することができます。また、各チャンネルはエッジ、又はレベルをモニタするよう個々にプログラムすることができます。

機器のシステムタイマーは指定された時間間隔でイベントを発動させることができます。例えば、0.005 秒間隔のタイマーにセットされた TRIG イベントソースは 200 Hz 周期の内部トリガを発生します。

前述したように、イベント条件はトリガバスチャンネルエッジ又はレベル、デジタル I/O チャンネルエッジ、レベル、タイマー時間間隔の任意の組み合わせを設定することができます。複数の ARM イベントソースが指定される場合、次のように論理的に組み合わされます。

ARM event = [(Timer tick event) AND (Digital I/O event) AND (Trigger Busevent)]

複数の TRIG イベントソースが指定される場合、次のように論理的に組み合わされます。

TRIG event = [(Timer tick event) AND (Digital I/O event) AND (Trigger Busevent)]

各デジタルハードウェアポート内でもまた特定のチャンネルに対して複数のチャンネルと複数の条件を指定することができます。その場合、次のように論理的に組み合わされます。

Digital I/O event = (Ch 7 events) AND (Ch 6 events) ... AND (Ch 0 events)

最後に、各チャンネル内でのよっつのイベント条件、Pos. Edge, Neg. Edge, Pos. Level, Neg. Level（上がりエッジ、下りエッジ、正相レベル、負相レベル）は OR で組み合わされます。

例えば、デジタル I/O イベントがチャンネル 3 の正相レベル、チャンネル 6 の上がりエッジ、そしてチャンネル 6 の下りエッジの組み合わせと定義された場合、チャンネル 3 の正相レベルがチャンネル 6 の上がりか下りのエッジと同時に起こった時にイベントが満たされることになります。ここではデジタル I/O イベント構成が例として出されましたが、トリガバスイベント構成も同じように働きます。

ひとくちメモ トリガモデルシステムの高い柔軟性は、高度に特殊なトリガ条件の作成を可能にすることによって、強力なアプリケーションツールとなり得ます。しかし、反面、実際には満たすことの難しいトリガ条件の作成を許すことにもなります。例えば、二つのエッジが同時に起こったと認識するにはお互いの時間差が 25 ns 未満である必要があります。同じように、タイマーソースはデジタルハードウェアエッジ条件と組み合わせるべきではありません。

測定性能の最適化

Section 3の *サンプリングレート・ノイズ*性能の項で説明された様に、EX1048 は 1 kHz未満のサンプリングレートにおいて複数サンプルの平均化という作業を行います。こうすることによって希望のデータ出力レートを維持しながらノイズ性能を高めることができます。但し、これが起こるにはトリガモデルが特定の形に設定されている必要があります。厳密には、TRIGイベントソースはTimer（タイマー）に設定されている必要があるのです。EX1048 がトリガイベントのレートとタイミングを正確に予測できるのはこの時のみなので、この設定はとても重要な要素となります。つまり、Timer設定では機器はデータの収集に掛かる時間が明らかなので、サンプルを幾つまで平均すればまだ全スキャンリストをカバーすることが出来るか、ということも予測できます。他のトリガソースを使用している場合は、トリガはいつでも起こりえます。その場合、EX1048 はスキャンリストを必要最低限の時間で完了させ、平均化の作業は行いません。

この為、トリガモデルの設定に、デジタル I/O ポート及びトリガバスを使用する必要がある場合は、TRIG ソースを Timer に設定して、必要とされているデジタルイベントを ARM ソースとして設定する方が良い結果を得ることが出来ます。TRIG ソースが Timer 以外に設定する必要がある例としては、二段階のデジタルイベントでトリガする必要がある時位です。例としては、デジタル I/O チャンネルレベルに対してアームが行われ、トリガバスチャンネルエッジにトリガされている時です。

最高トリガレート

TRIG イベントソースをデジタル I/O チャンネルレベルやトリガバスチャンネルエッジなどの外部イベントに設定できるという事は、測定を外部のイベントに正確に同期できる、強力なテスト機能となります。但し、外部イベントは場合によっては EX1048 が DEVICE レイヤーを正しく完全に処理するよりずっと速いレートで発生することがありえます。この為、機器が過負荷状態になるのを防ぐ為に、トリガモデルはペースタイマーを内蔵しています。いったんトリガイベント条件が満たされると、このタイマーは 990 µs の間トリガイベントの認識を無効にします。この 990 µs 区間のトリガイベントはバッファされず、完全に無視されます。これはトリガモデルが ARM イベント待ちでない時は新たなる ARM イベントが無視されることと似ています。この動作方式の為、1 kHz を超える外部パルスのバーストでトリガさせると、EX1048 の測定はトリガイベント数だけでなく最高サンプリングレートを大幅に下回る読み込みレートになってしまう事があります。

例として、以下の様に構成されたトリガモデルを見てみましょう。

Init Continuous	Disabled
Timer Interval	0.1
Arm Source	Immediate
Arm Count	1
Arm Delay	0
Trig Source	DIO Channel 0 Pos. Edge
Trig Count	1
Trig Delay	0

いったん開始されると、EX1048 は 1.5 kHz のレートで 100 回の上がりエッジのバーストによって駆動します。1.5 kHz ではトリガイベントの間隔は 667 µs です。これは 990 µs より短い時間なので、トリガイベントは 1 つおきに無視されます。その為、このバーストは 750 Hz のレートで 50 回の読み込みとして測定されます。ご覧の通り、実際の読み取りデータ出力は最高値の 1 kHz より大幅に低い値となっています。最高の出力レートが必要な場合は TRIG イベントは Immediate（即時動作）か Timer（時間間隔は 0.001 で）動作にセットする必要があります。

第 5 章

ウェブインターフェイス操作

はじめに

EX1048 はプログラミングを必要とせず、簡単にリモート操作できるウェブインターフェイスを内蔵しています。ウェブインターフェイスでは、殆どの EX1048 の設定変更やデータ収集設定が可能なだけでなく、ネットワークパラメータを確認、設定をする場所でもあります。

ウェブインターフェイスを開く

EX1048 に割り当てられた IP アドレスをウェブブラウザに入力してください。

基本ウェブ操作

EX1048 に接続すると、まず最初に計器のホーム画面が現れます。この画面には以下の計器特有の情報が表示されます。

- シリアルナンバー
- MAC アドレス
- ファームウェアのバージョン情報
- 最後に完全校正（Full Calibration）をした日付
- 不揮発性メモリ上の自己校正（Self Calibration）情報の有無
- エラー状況

他の操作画面からこの画面を参照するときはコマンドメニュー上の **EX1048 Main** リンクをクリックしてください。

ウェブインターフェイスで操作可能な設定、データ収集、データ所得機能はすべて画面左に表示された EX1048 コマンドメニューを通して利用可能です。コマンドメニュー上の機能は大きく三種類に分けられます。

状態表示 この画面は、動作状況や情報の表示をします。命令、入力ともに受け付けません。**EX1048 Main** などは状態表示画面に分類されます。

実行画面 この画面では、測定器に対して命令を実行できます。パラメータの入力は行いません。**Trigger…Initialize** などは実行画面に分類されます。

入力画面 この画面は、測定器の設定を表示、変更できます。**Scan List…Configuration** などは入力画面に分類されます。

EX1048 の入力画面は幾つか共通する特質を持っています。

- 必ず *Apply*、*Cancel* の 2 つの標準ボタンを備えています。
- 画面ロードとともに現在のパラメータ、または選択を表示します。
- 画面ロード後の変更は *Apply* ボタンをクリックするまで適用されません。*Apply* ボタンをクリックすると画面は最新の情報に更新されます。
- 画面ロード後の変更は *Cancel* ボタンをクリックすることによって破棄できます。パラメータ、選択は最後に設定した値に戻ります。変更を適用する前に他の画面に移行しても同じ効果が得られます。
- パラメータ画面のナビゲーションは Tab キーで行います。Enter キーは *Apply* ボタンと同じ機能がありますので、ナビゲーションには使えません。

TRIGGER（トリガ）メニュー

Trigger メニューでは EX1048 のトリガシステムの設定、作動が可能です。

Configuration（設定）

この入力画面では、EX1048 のアーム、及びトリガソースの設定をします。この画面は、ページ訪問毎に階層型メニューの表示画面を記憶しておきます。こうすることによって、トリガシステムの必要な部分だけを表示し、設定オプションのナビゲーションを容易にします。トリガについての詳しい説明は Section 4をご参照ください。この画面での設定の大部分はチェックボックスによる機能のオン・オフで行います。残りは各パラメータ値の入力で、その範囲は以下の通りです。

パラメータ	最低値	最高値	精度
Timer Interval (s)	0.001	4294	0.000001
Arm Count	1	$(2^{31}-1)$	1
Arm Delay (s)	0	4294	0.000001
Trig Count	1	$(2^{31}-1)$	1
Trig Delay (s)	0	4294	0.000001

表5-1: トリガ設定パラメータの範囲

また、この画面は、二つの標準ボタンのほかに *Default* ボタンを備えています。このボタンを使うことによって、トリガ設定パラメータをクリック一つでリセット値に戻すことができます。これらの選択は *Apply* ボタンをクリックするまで適用されないので、注意してください。

Initialize（開始）

この実行画面では、トリガモデルを IDLE レイヤーから ARM レイヤーへ移行し、データの収集を始めます。この命令は、FIFO リーディングメモリを消去し、全ての限界値表示をリセットします。

Abort（中止）

この実行画面では、現在のデータ収集作業を中止し、トリガモデルを IDLE レイヤーに戻します。いったん収集が始まると、EX1048 はほとんどの命令を受け付けなくなります。設定変更のためにデータ収集をいったん中止する場合、**Abort** 命令が必要になります。既に収集したデータは **Abort** では影響されません。

Software Arm（ソフトウェア・アーム）

この実行画面では、トリガモデルのソフトウェア・アームを実行できます。ソフトウェア・アームは常に ARM ソースとして適用されており、設定されている ARM ソースのバイパスすることができます。ソフトウェア・アームは、命令が出された時点で適用され、システムが ARM イベント待ちの状態でない場合は無視されます。ソフトウェア・アームの実行は、ARM ソース設定には影響されません。

Software Trigger（ソフトウェア・トリガ）

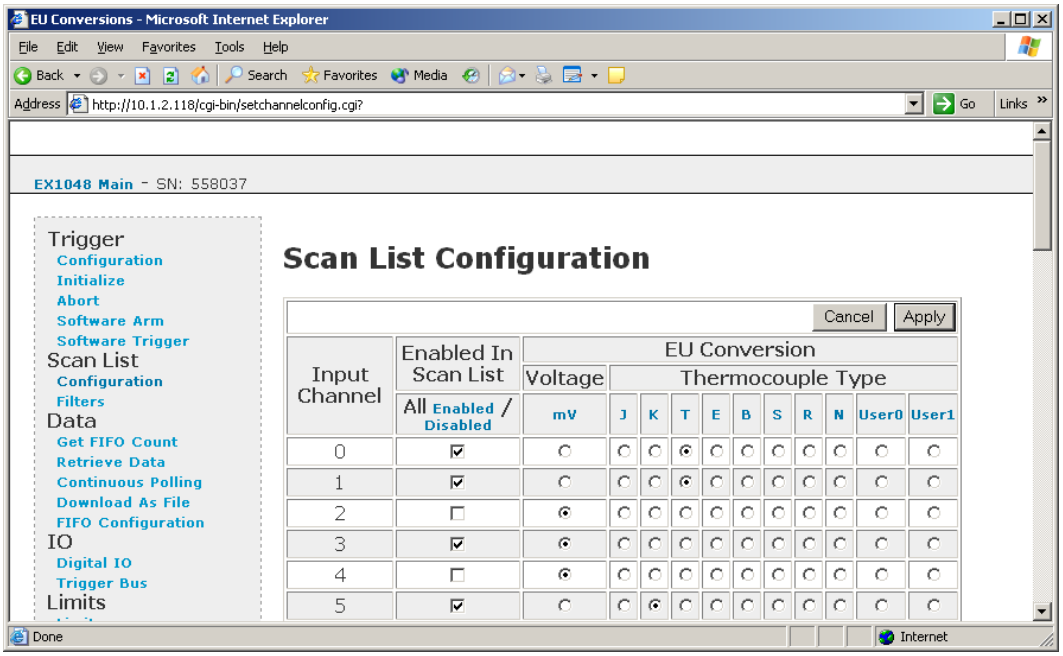
この実行画面では、トリガモデルのソフトウェア・トリガを実行できます。。ソフトウェア・トリガは常に TRIG ソースとして適用されており、設定されている TRIG ソースをバイパスすることができます。ソフトウェア・トリガは、命令が出された時点で適用され、システムが TRIG イベント待ちでない場合は無視されます。ソフトウェア・トリガの実行は、TRIG ソース設定には影響されません。

SCAN LIST（スキャンリスト）メニュー

Scan List メニューでは EX1048 の入力チャンネルのスキャンリスト上の有無、EU 変換、ハードウェアフィルタの設定ができます。

Configuration（設定）

この入力画面ではチャンネルごとのチャンネルスキャンリストと EU（工学単位）変換設定ができます。



画像5-1: SCAN LIST CONFIGURATION

上の画面では現在チャンネル 0, 1, 3, 5 がスキャンリスト上に選ばれ、それぞれ T, T, mV, K の EU 変換に設定されています。

スキャンリスト上のチャンネルは対応するチェックボックスにチェックを入れることによって選びます。また、*Enabled/Disabled* リンクをクリックすることによってすべての

チャンネルの一斉チャック・アンチェックが可能です。同様にチャンネルごとの EU 変換は対応するラジオボタンを種類で選択します。これらの選択は *Apply* ボタンをクリックするまで適用されないので､注意してください。

Filters（フィルタ）

この入力画面ではチャンネルごとのハードウェアフィルタを設定します。フィルタはチャンネルごとに望みの周波数に対応するラジオボタンをクリックすることによって設定できます。また、それぞれのリンクをクリックすることによって、すべてのチャンネルを 1 つの周波数に固定することもできます。これらの選択は *Apply* ボタンをクリックするまで適用されないので､注意してください。

DATA（データ）メニュー

Data メニューでは EX1048 からデータを取り込むとともに取得するデータの形式を設定できます。

Get FIFO Count（FIFO カウント取得）

この状態表示画面は、まだ回収していない収集データの残りページ（スキャン）数を表示します。

Retrieve Data（データ読み出し）

この実行画面では、１ページ（スキャン）分の収集データを FIFO（First-In First-Out）形式で回収、表示します。FIFO が空である状態でこの画面を呼び出した場合、エラーメッセージが表示されます。将来のデータ収集時に最大限のリーディングバッファ容量を確保する為、FIFO メモリ内のデータは回収と共に消去されます。

Continuous Polling（連続ポーリング）

この実行画面では、EX1048 を連続的にトリガしながら、３秒に１回の間隔で１ページ（スキャン）分のデータを回収します。データモニタリング全般、導入時のデバッグ、また機器のコントロールに慣れ親しむときに便利になります。ただ、データ収集のためにトリガ設定を変更する為、トリガシステムのテストには使えません。連続ポーリングを停止するときは、*Stop Polling（ポーリングの停止）*ボタンか他画面へのリンクをクリックしてください。*Stop Polling* ボタンを利用した場合は、事前のデータ表示を維持します。

ひとくちメモ **Continuous Polling** 画面の使用はトリガ設定を変更します。以前に入力した設定値は再入力する必要があります。

Download As File（ファイルとしてダウンロード）

この入力画面では、FIFO メモリの内容の一部、又は全部をファイルとしてダウンロードできます。対応しているファイルフォーマットは、CSV、Raw Binary、そして MATLAB です。初期設定では、すべてのスキャンがダウンロードされます。特定のページ数が必要な場合は、**ALL** ボックスをアンチェックし、スキャン数を入力してください。将来のデータ収集時に最大限のリーディングバッファ容量を確保する為、FIFO メモリ内のデータは回収と共に消去されます。

FIFO Configuration（FIFO 設定）

この入力画面では、FIFO メモリのデータ形式、及びオーバーフロー時の対応を設定できます。

Report CJC Temperatures 設定では、収集データ内に CJC（冷接点補償）温度の報告を設定します。この設定は、収集データの表示を設定するものであり、実際の測定に影響あるものではありません。CJC 温度の測定は、この設定に関わらず、スキャン毎にされます。報告される CJC 温度は、**Temperature Units** の選択に関わらず ºC の単位で表示されます。

Report Timestamps 設定では、収集データ内にチャンネル毎のデルタタイムスタンプの報告を設定します。このタイムスタンプは、特定のチャンネル測定のスキャン開始からの測定までの時間を 100 ns の精度で表しています。

Temperature Units 設定では、入力チャンネルデータの戻り値の温度単位（ºC か ºF）を設定します。この設定は、報告される CJC 温度データや mV に工学単位変換設定された入力チャンネルのデータには影響しません。

Blocking Mode 設定では、スキャン時にリーディングバッファがいっぱいになったときのシステムの対応を選ぶことができます。ブロッキングモードが適用されている場合は、それ以降の測定データは放棄されます。ブロッキングモードが適用されていない場合は、新しいデータはバッファ内のデータを古い順から上書きしていきます。

IO（入出力）メニュー

IO メニューでは EX1048 のデジタル I/O ポート及びトリガバスよりデータの取り込み、出力ができます。

Digital IO（デジタル I/O）

この入力画面では、8 チャンネルデジタル I/O ポートの状況を監視し、汎用出力デバイスとして制御できます。各 DIO（デジタル I/O）チャンネル は、出力時の出力機能と取るべきスタティックレベル、そしてパルス動作を個々にプログラムできます。

Input State 状況表示画面では、アクセス、更新時の DIO チャンネルで測定されたデジタル値（0 か 1）を表示します。この情報は、入力値移行時やタイマーでは自動更新しません。その為、静止した、またはゆっくりした変化に対してのみ利用してください。入力監視機能は **Output Enable** 操作に関わらず作動します。

Output State 設定では、各 DIO チャンネルの出力値をプルダウンメニューで選択できます。この設定はチャンネルが出力として設定されたときのチャンネルの値を表しています。チャンネルが適用されていない場合は、この選択は効果ありません。

Output Enable 操作では、各 DIO チャンネルを出力として適用できます。適用されると、チャンネルは **Output State** 設定で選択された値を持ちます。出力として適用されていない場合は、チャンネルは 3 ステート状態になります。

ひとくちメモ DIO チャンネルを出力として設定する前に、チャンネルにアクティブな電圧ドライバが他に繋がっていないことを確認してください。機器の誤作動や破損に繋がる恐れがあります。

Pulse 操作では、1 µs のシングルショットパルスを生成できます。パルスの形状は、そのチャンネルにプログラムされたスタティックレベルに依存します。チャンネルのスタティックレベルが high にプログラムされている場合、パルスは low になります。チャンネルのスタティックレベルが low にプログラムされている場合は、パルスは high になります。各パルスを出力する毎に、個別に命令を実行する必要があります。この操作は、チャンネルが出力として設定されている場合のみ効果があります。普通、パルス操作は

チャンネルのスタティックレベルが既に定義されいる状態で使われます。しかし、出力値の変更とパルスの出力の両方が同時に適用された場合、チャンネルはまず出力を新しい値に変更してから、適切なパルスを出力します。

Limit Event 状態表示では、DIO チャンネルの限界値評価との関連付けの有無（Y か N）を表示します。適用されている場合、チャンネルの出力値は **DIO Limit Events** 画面で設定された限界値条件を第一に管理されます。ここでこの情報が表示されるのは、DIO Limit Event として関連付けされたチャンネルは一般的に直接制御用出力として使用されないためです。しかし、直接出力制御が無効になっている訳ではないので、これによって関連チャンネルを非同期的にリセットすることもできます。

Trigger Bus（トリガバス）

この入力画面では、8 チャンネルトリガバス（VTB）の状況を監視し、汎用出力デバイスとして制御できます。各 VTB チャンネル は、出力時の出力機能と取るべきスタティックレベル、そしてパルス動作に対して個々にプログラム可能です。

Input State 状況表示画面では、アクセス、更新時の VTB チャンネルの測定されたデジタル値（0 か 1）を表示します。この情報は、入力値移行時やタイマーでは自動更新しません。その為、静止した、またはゆるい変化に対してのみ利用してください。入力監視機能は **Output Enable** 操作に関わらず作動します。

Output State 設定では、各 VTB チャンネルの出力値をプルダウンメニューで選択できます。この設定はチャンネルが出力として設定されたときのチャンネルの値を表しています。チャンネルが適用されていない場合は、この選択は効果ありません。

Output Enable 操作では、各 VTB チャンネルを出力として適用できます。適用されると、チャンネルは **Output State** 設定で選択された値を持ちます。出力として適用されていない場合は、チャンネルは 3 ステート状態になります。

ひとくちメモ	VTB チャンネルを出力として設定する前に、チャンネルにアクティブな電圧ドライバが他に繋がっていないことを確認してください。機器の誤作動や破損に繋がる恐れがあります。

Pulse 操作では、1 µs のシングルショットパルスを生成できます。パルスの形状は、そのチャンネルにプログラムされたスタティックレベルに依存します。チャンネルのスタティックレベルが high にプログラムされている場合、パルスは low になります。チャンネルのスタティックレベルが low にプログラムされている場合は、パルスは high になります。各パルスを出力する毎に、個別に命令を実行する必要があります。この操作は、チャンネルが出力として設定されている場合のみ効果があります。通常、パルス操作はチャンネルのスタティックレベルが既に定義されいる状態で使われます。しかし、出力値の変更とパルスの出力の両方が同時に適用された場合、チャンネルはまず出力を新しい値に変更してから、適切なパルスを出力します。

LIMITS（限界値）メニュー

Limits メニューでは、EX1048 の限界値評価システムの設定ができます。

Limits（限界値）

この入力画面では、二つのプログラマブルリミットセットの値を定義することができます。これらの限界値、リミットセット 0 とリミットセット 1 は、チャンネル毎にプログ

ラム可能です。画面では一度に 1 組の限界値が表示されます。リミットセット間の切替は **Set** 設定でプルダウンメニューを介してできます。

リミットセット 0 の値は、初期設定で、各チャンネルの工学単位変換タイプと温度単位設定を元に自動的に決められます。つまり、リミットセット 0 の上限値と下限値は、表 3-2で表記されているEX1048 の測定範囲の上限値と下限値に設定されます。例えば、入力チャンネル 0 がE熱電対とºCの単位に設定されている場合、上限値と下限値はそれぞれ+900 と-200 となります。この自動設定を無効にし、チャンネル毎に手動で上限値と下限値を入力するには、**Manual**チェックボックスにチェックを入れ、新しい値を入力してください。値は、10 進表記法、科学小数点法のどちらでも入力できますが、表示は常に科学小数点法でされます。リミットセット 1 は、手動モードでのみ働きます。

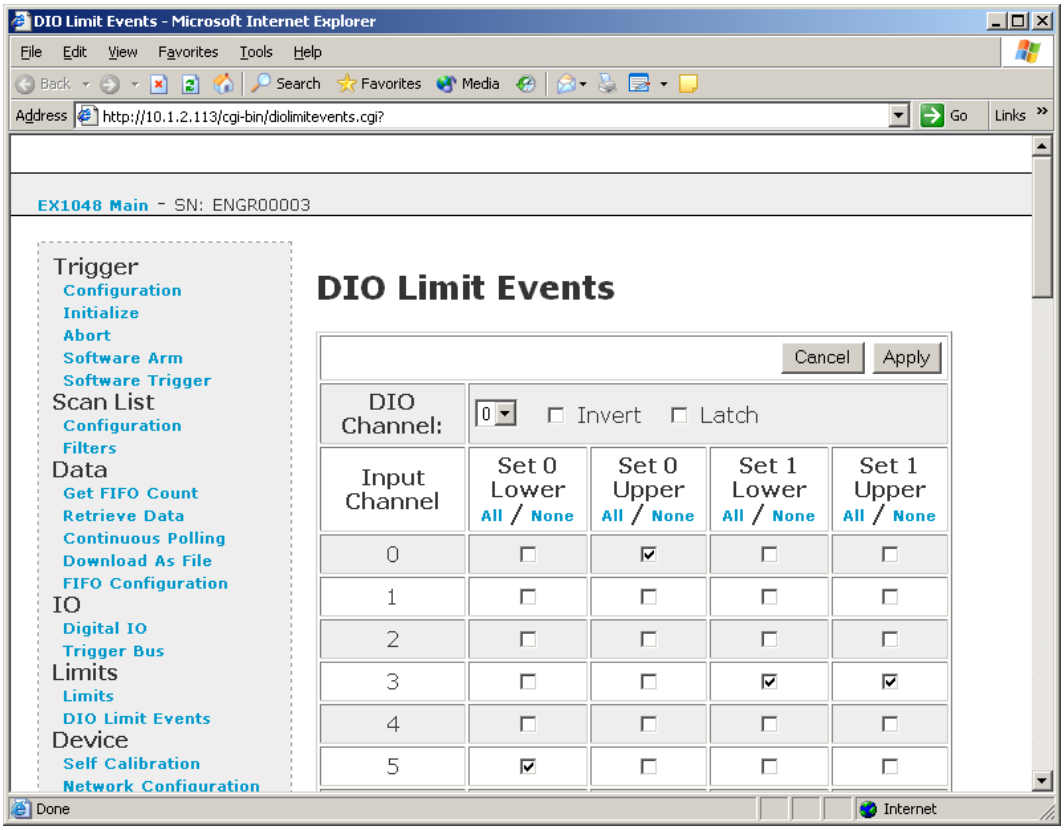
ひとくちメモ マニュアルモードで動作しているリミットセットは工学単位変換タイプや温度単位設定の変更に合わせて変換されることはありません。間違えの無い様、限界値は工学単位変換タイプや温度単位の設定が終わってから行ってください。また、設定が変更された場合は限界値も忘れずに変更してください。

DIO Limit Events（DIO リミットイベント）

この入力画面では、ひとつ、または複数の限界値評価をデジタル I/O ポートの動作に関連付けることができます。普通、リミットセット 0 の評価結果はフロントパネル LED に表示され、両方のリミットセット評価結果は機器のドライバより入手できます。DIO リミットイベントメカニズムは加えて、ひとつまたは複数の入力チャンネルの限界値評価より DIO チャンネルを正確に操作することを可能にします。

DIO リミットイベントはチャンネル毎にプログラム可能です。つまり、各 8 つの DIO チャンネルはそれぞれ個別の動作特性を持たせることが出来るのです。画面では一度にひとつの DIO チャンネルのリミットイベント操作画面しか表示されません。DIO チャンネル間の移動は **DIO Channel** プルダウンメニューで行います。

DIOチャンネルを特定の入力チャンネルの特定の限界値条件に結びつける作業は適切なチェックボックスをクリックすることによってなされます。複数個の選択も可能で、論理的にはORで結び付けられます。例えば、画像 5-2ではDIOチャンネル 0 が入力チャンネル 0、3、5 の限界値条件と結び付けられています。この場合、DIOチャンネル 0 は指定された 4 つの限界値のいずれかひとつが満たされれば状態を移行することになります。



画像5-2: DIO LIMIT EVENTS

複雑な設定の入力を容易にするために、*All* 及び *None* のリンクで全入力チャンネルのリミットセット値の有効／無効設定を行えます。これら、リンクは *Apply* ボタンをクリックするまで適用されないので､注意してください。

DIO リミットイベントとスキャンリストの設定は個別に行われる作業です。つまり、限界値条件を **DIO Limit Events** 画面で指定するだけではスキャンリストに適用されることにはなりません。同じく、入力チャンネルをスキャンリストから削除しても自動的に限界値条件との関連付けがなくなることもありません。スキャンリスト上にないチャンネルへ関連付けられた DIO リミットイベントは評価の時には無視されるので、誤動作の原因となることはありません。

リミットイベントとして関連付けされた場合、DIO チャンネルは新しい収集の始めに初期化されます。そして、そのステートはプログラムされた限界値評価に応じてスキャン毎にアップデートされます。初期設定では、DIO の初期化された状態は low ですが、**Invert** ボックスにチェックを入れることによってチャンネルごとに high にセットできます。同じ様に、各チャンネルの初期設定値は非ラッチモードですが、**Latch** ボックスにチェックを入れることによってチャンネルごとにラッチモードにすることができます。ラッチモードでは、将来の限界値評価結果に関係なく、新しい収集シーケンスで初期化されるまで、初期化状態からの移行は保持されます。

DEVICE（デバイス）メニュー

Device メニューでは EX1048 の詳細設定ができます。

Self Calibration（自己校正）

この実行画面では、自己校正関連の操作ができます。Section 3の *自己校正*で説明されている様に、EX1048 の測定性能は定期的な自己校正をすることによって、大幅に高めることができます。自己校正作業は、短時間で終わる上、入力端子を切断したり、外部機器を利用する必要がないので、簡単に何度でも行えます。この自己校正機能を十分に引き出すには、自己校正定数がどうやって生成、使用、保存されるかを知っておく必要があります。

自己校正は、完全校正（Full Calibration）時に生成された不揮発性メモリ内の校正定数を上書き、変更、置き換えることはありません。代わりに、測定計算に完全校正定数が適用された後、これを補足する付加的な校正定数を生成します。自己校正が実行されると、ここで計算されたデータは揮発性メモリ内に置かれます。このデータは “current self cal data”（現在自己校正データ）と呼ばれ、機器によって自動的に使われます。使用する際に特別に命令を出す必要はありません。ただ、揮発性メモリ内に置かれている為、機器のリセット、又は電源の再投入と共に消去されます。また、“current self cal data”を不揮発性メモリ内に保存することはできます。

自己校正を実行するには、まず機器のロックを取得する必要があります。ロックに関する詳しい情報は、この章の **Locking（ロック）**メニューの説明をご参照ください。ロックが取得されていない状態で自己校正を試みると、エラーメッセージが表示されます。この画面上の*Perform Self Calibration*ボタンをクリックすると自己校正が始まります。いったん自己校正を始めると、校正処理の実行確認が表示され、この後、定期的に更新しながら進行状態をパーセント値で表示します。この値が 100%になると、自己校正は完了です。

機器のウォームアップ中にユーザーが誤って自己校正を実行することを防ぐ為、EX1048 が 60 分間継続的にオンになっていなかった場合は、警告メッセージが表示され、自己校正は実行されません。これは単なる警告であり、この時表示される *Override Self Calibration Uptime* ボタンをクリックすることによって無視できます。機器が完全にウォームアップする前に自己校正を実行すると、測定性能を下げる可能性があるので、一般的にはあまりお勧めできません。ただ、機器が既にウォームアップされ、一時的な電源の再投入、又は再起動をしただけの場合は、警告を無視してくださっても結構です。

上記で説明されたように、自己校正処理を実行すると計算されたデータは揮発性メモリ内に“current self cal data”として置かれます。いったんここに置かれると、このデータは *Clear Current Self Cal Data* ボタンで消去できます。この操作は、揮発性メモリ内のデータは消去しますが、不揮発性メモリ内の自己校正データは影響されません。

いつでも可能なこの自己校正ですが、自己校正が必要でありながら、不揮発性のデータが要求されるケースもあります。*Store Current Self Cal Data As Nonvolatile* ボタンは current self cal data を不揮発性メモリ内に保存し、機器の電源再投入、また再起動と共にロードできます。current self cal data が存在しない状態でこのボタンが押された場合は、エラーが発生します。不揮発性自己校正データの保存は出荷時の校正設定からの永久的な（取り消しは可能）変更をもたらす行為である為、 その存在は **EX1048 Main** 及び **Self Calibration** 画面に表示されます。

以前に保存された不揮発性自己校正データは *Clear Nonvolatile Self Cal Data* ボタンにより消去できます。この作業は、不揮発性メモリ内の自己校正データだけを消去する物であり、current self cal data は影響されません。不揮発性自己校正データが存在しない状態でこのボタンが押された場合は、エラーが発生します。逆に、以前保存された不揮発性自己校正データは *Load Nonvolatile Self Cal Data* ボタンで current self cal data としてロード

できます。既に current self cal data が存在する場合は、そのまま上書きされますので、事前に消去する必要はありません。不揮発性自己校正データが存在しない状態でこのボタンが押された場合は、エラーが発生します。

current self cal data、及び nonvolatile self cal data（不揮発性自己校正データ）の存在は、自己校正の実行には何の影響もありません。自己校正処理を実行すると、元からあった current self cal data はそのまま上書きされます。新しく自己校正をする前に *Clear Nonvolatile Self Cal Data* ボタンをクリックする必要はありません。同じく、以前に保存された不揮発性自己校正データの存在は、current self cal data の不揮発性メモリへの保存には影響ありません。元からあったデータはそのまま上書きされますので、事前に消去する必要はありません。

ひとくちメモ 自己校正プロセスの一環として、EX1048 はいったんリセット作業を行います。この時機器の設定パラメータはすべて初期設定値に戻されます。

Network Configuration（ネットワーク設定）

この入力画面では EX1048 のネットワーク設定の変更ができます。EX1048 はデフォルトで DHCP サーバ経由で IP アドレスを取得するよう設定されています。DHCP サーバが発見できない場合は、20 秒のタイムアウト後、AutoIP を利用して IP アドレス取得をします。

AutoIP は 169.254.X.Y の範囲で未使用の IP アドレスを割り当てる機能です。ここでは X は 1 – 254、Y は 0 – 255 の範囲を示します。機器はまず自分の MAC アドレスを元に最後から二番目のオクテットを X、最後のオクッテットを Y とした上で、169.254.X.Y のアドレスを割り当てます。ここで X に相当するオクテットが 0 の場合は 1 が、Y に相当するオクテットが 255 の場合は 254 が代わりに割り当てられます。このアドレスが既に使用されている場合は、割り当てのされていないアドレスを擬似乱数的に探し出します。

このAutoIP機能の例が、表 5-2です。ここでは、各MACアドレスに対応した初期割り当てアドレスを表示しています。

MAC アドレス	AutoIP 初期アドレス
00:0D:3F:01:00:01	169.254.1.1
00:0D:3F:01:01:01	169.254.1.1
00:0D:3F:01:A3:28	169.254.163.40
00:0D:3F:01:FE:FE	169.254.254.254
00:0D:3F:01:FF:FE	169.254.254.254

表5-2: AUTOIP 初期割り当てアドレス

固定 IP アドレスの割り当てが好まれる場合は、この画面で割り当てる事もできます。手順は、DHCP と書いてある箇所の右にある *Disabled* のラジオボタンをクリックし、IP Address の欄に固定 IP アドレスを入力してください。変更を有効にするには **Hard Reboot** が必要です。また、ネットワーク設定は不揮発性メモリに保存されていますので、**Reset Device、Hard Reboot** 命令には影響されません。

しかし、固定 IP アドレスを最も便利に利用する方法は、機器のほうでは DHCP を有効にしておき、会社の DHCP サーバ内で機器の IP アドレスを予約しておくという方法です。この方法では、予約 IP アドレスは DHCP サーバ内で機器の MAC アドレスと関連付けされ、EX1048 で設定しなくとも機器の起動と共に自動で割り当てられます。DHCP サーバで集中型 IP 配置をすることによって、管理を一箇所にまとめるだけでなく、複

数の機器に同じ IP を割り当てる危険も回避できます。サーバの設定は会社の IT 事業部に相談してください。

EX1048 では VXI-11 Device Discovery にも対応しています。これを使えば MAC アドレス、IP アドレスの予備情報がなくともブロードキャストを利用してローカルネットワーク内すべての EX1048 を発見できます。

リセットボタン

ネットワーク設定を間違った場合、またはネットワーク設定が不明な場合、EX1048 のリア（背面）パネル部にあるリセットボタンを使ってネットワーク設定を初期設定値に戻すことができます。ネットワークのリセットの手順は以下のとおりです。

6) EX1048 の電源を切ります。
7) リセットボタンを押し続けます。
8) ボタンを押したまま、電源を入れます。
9) そのまま 30 秒、リセットボタンを押したままでいます。
10) リセットボタンを放します。

EX1048 は直前のネットワーク設定でなく初期設定の値（DHCP）を使いますが、それ以外は普通に起動します。

Time Configuration（時間設定）

この入力画面では、EX1048 の時間設定ができます。EX1048 は出荷状態では SNTP (Simple Network Time Protocol) より時間を受け取るよう設定されています。**Zone** コントロールでは機器の現在位置の時間帯をプルダウンメニューで選択できます。新しい選択をした場合はリブートが必要となります。また、ユーザーが手動で時間を設定することもできます。これは機器がインターネットにつなげられないようなネットワーク環境にある場合に必要になります。時間の手動設定をするには **Source** 項目の **Manual** というラジオボタンを選択し、日付と時間を画面で指定した通りに入力してください。手動での時間設定は **Zone** コントロールに影響されない為、機器のリブートは必要ありません。しかし、手動設定時間は揮発性メモリ内にありますので、機器のリブート、及び電源の再投入の再に入力をしなおす必要があります。**Reset Device** 命令には影響されません。

Upgrade Firmware（ファームウェアの更新）

この実行画面では EX1048 の内蔵ファームウェアをアップグレードできます。アップグレード作業をする前に、まず VTI Instruments より最新のファームウェアイメージを手に入れ、EX1048 につながっているコンピュータからアクセス可能な場所にある必要があります。特にそうでない旨書いてない限り、EX1048 の校正を変更することはありません。

ひとくちメモ ファームウェアのアップグレード中は EX1048 の電源を切らないでください。アップグレード作業中に電源が切れた場合、機器が動作不能になり、修理が必要となる場合があります。リスク防止の為、無停電電源装置をご利用ください。

ファームウェアアップグレードを行う為には次の手順を踏んでください。

1) **Hard Reboot**、又は電源を再投入します。
2) 内蔵ウェブインターフェイス経由で EX1048 につなぎます。
3) **Upgrade Firmware** リンクをクリックします。
4) *Proceed* ボタンをクリックします。

5) *Browse* ボタンをクリックし、機器にアップロードするファームウェアイメージを選択します。
6) *Upload* ボタンをクリックし、アップグレード作業を始めます。

ひとくちメモ いったんアップグレード作業が始まったら、機器への接続を試みないでください。再接続は、下で説明されたとおりに作業が終わるまで待ってください。

アップグレード作業中はフロントパネル部の LED が 6 つ、CH0 から CH5 まで順番に点灯し、作業状況を表します。最後に全 48 チャンネル分の LED が一回点灯し、アップグレードの終了と再起動を示します。この作業は約 5 分ほどかかります。

一旦、機器が再起動したらつなぎ直し、**EX1048 Main** 画面でファームウェアバージョンが正しく更新されていることを確認してください。

Reset Device（デバイスリセット）

この実行画面では、EX1048 のデータ収集設定パラメータをすべて初期設定値に戻すことができます。これは、一般的に新しいテストシーケンスを始める前に、機器を既知の状態に戻す為に使います。影響を受ける各パラメータとリセット値は 表 6-1を参照ください。自己校正データにおいては、機器のリセットは、現在の自己校正データを消去し、存在する場合は不揮発性メモリ上の自己校正データをロードします。このデータの作成、利用に関する詳しい情報は、この章の **Self Calibration（自己校正）** 画面の説明を参照ください。

ひとくちメモ 機器のリセットは FIFO リーディングメモリを消去します。収集したデータは、この命令を実行する前に FIFO より回収する必要があります。

Hard Reboot（再起動）

この実行画面では、機器の完全な再起動、つまり電源の再投入と同等の操作を行う事ができます。これは、一般的にネットワーク設定や時間設定を変更したときに使います。また、ファームウェアのアップグレードを行う前にも再起動をすることをお勧めします。最後に、機器のファームウェア状態がおかしくなった場合、リカバリ作業をする為にも使えます。

LOCKING（ロック）メニュー

Locking メニューでは、EX1048 のデータ収集システム上のロックの取得、解除、切断ができます。

Lock（ロック）

この実行画面を利用すると、ロックの取得を試みます。保護された作業を望まれる場合は、ロックを使用する習慣をつけておくと良いでしょう。また、自己校正を実行するときもロックの取得が必要となります。EX1048 が既に他のホストによってロックされている場合は、ロック作業は失敗し、エラーメッセージが表示されます。

Unlock（ロック解除）

この実行画面を利用すると、前に取得したロックの解除を試みます。ロックの解除は、前にロックを取得したホストにしかできません。ホスト以外の者がロック解除を試みた場合、解除は失敗し、エラーメッセージが表示されます。また、機器がロック状態でない時にロック解除を試みた場合も、エラーメッセージが表示されます。

Break Lock（ロックの切断）

この実行画面を利用すると、ロックを取得したホストに関わらず、ロックを切断します。切断をしてもロックの取得は自動的にはされません。ロックの取得が望まれる場合は、別に **Lock** 命令を実行する必要があります。

ADVANCED MENU（高度設定）

高度設定メニューでは、EX1048 でユーザー定義の熱電対多項式や冷接点補償温度を設定することができます。標準的な熱電対や一般的なアプリケーションで、これらの機能が必要になることはありません。

User Conversions（ユーザー定義変換）

この入力画面では、熱電対多項式にユーザー定義の係数を入力できます。係数は 2 セットまで入力可能です。詳細は Section 3の *ユーザー定義変換式*を参照ください。ユーザー定義の係数は、入力しただけでは測定に使用されません。使用するには**Scan List…Configuration**画面のEU ConversionでUser0 かUser1 を選択してください。

User CJC Temp（ユーザーCJC 温度）

この入力ページではユーザー定義の CJC（冷接点補償）温度を入力、適用できます。各チャンネルは、独自の値に対応付けられ、個々に適用させることができます。適用した場合、熱電対計算は内部測定された物ではなく、ユーザー定義の CJC 温度値を使って実行されます。また、外部 CJC 値の入力とその適用は個別のものであることを覚えておいてください。つまり、値の入力は、その使用を意味する訳ではなく、既に適用された値を無効にしても、その値が消去される訳ではありません。すべての入力は ºC の単位で行います。

ひとくちメモ	入力チャンネルデータには、外部 CJC 値が使用されている事は表示されないので、この機能を使う場合は、くれぐれも注意してください。

設定例

例 1

これは、アラームを使った静止モニタリングアプリケーションの一般的な設定シーケンスの例です。このアプリケーションでの必要条件は以下の通りです。

- 接続入力はチャンネル 0-4 に 5 つの J 熱電対、チャンネル 5-9 に 5 つの K 熱電対を使用します。
- 測定される温度環境は 4 Hz 周波数フィルタに適した変動の緩やかなものです。
- 測定データは ºF の単位が求められています。
- CJC 温度の報告が求められています。
- データは DIO チャンネル 0 が high になった時に 20 測定／秒で収集される必要があります。
- いったん測定が開始したら、中止命令が出るまで続けられます。
- アラーム表示には DIO チャンネル 1 が利用されます。いずれかの入力チャンネルが 100 ºF を超えた場合、high に移行し、その状態を保持する必要があります。

このアプリケーションの条件を満たす為には以下の手順を踏んでください。

スキャンリストの設定

Scan List…Configuration 画面でスキャンリストを設定します。前に設定したチャンネルを初期状態に戻す為、*Disabled* リンクをクリックします。クリアされたらチャンネル 0－9 にチェックを入れます。熱電対はチャンネル 0－4 は J 熱電対、チャンネル 5－9 は K 熱電対にそれぞれ対応するラジオボタンをクリックしてください。K.設定の変更を適用する為、*Apply* ボタンをクリックしてください。

フィルタ周波数の設定

Scan List…Filters 画面でフィルタを設定します。チャンネル 0－9 で 4 Hz のラジオボタンをクリックします。また、全適用チャンネルで同じフィルタ設定を利用する為、*4 Hz* リンクで一度に全チャンネルを設定することもできます。設定の変更を適用する為、*Apply* ボタンをクリックしてください。

FIFO の設定

Data…FIFO Configuration 画面で FIFO を設定します。**Temperature Units** で°F を選択します。**Report CJC Temperatures** ボックスにはチェックを入れます。設定の変更を適用する為、*Apply* ボタンをクリックしてください。

トリガモデルの設定

Trigger…Configuration 画面でトリガモデルを設定します。まずは *Default* ボタンをクリックします。Arm 枝を展開し、Arm Source の設定を表示します。ソースが選択されている場合は全部アンチェックしてください。また、Digital I/O 枝を展開し、DIO Channel 0 の Pos. Level を選択します。Trig Source では Timer を選択してください。**Timer Interval** は 0.05 と入力します。後は **Init Continuous** ボックスにチェックを入れます。設定の変更を適用する為、*Apply* ボタンをクリックしてください。

限界値設定

Limits…Limits 画面で限界値条件を設定します。プルダウンメニュー**Set** では Set 0 を選択します。チャンネル 0－9 に対しては **Manual** ボックスにチェックを入れ、上限値として 100 と入力します。設定の変更を適用する為、*Apply* ボタンをクリックしてください。これら限界値条件は **Limits…DIO Limit Events** 画面を通して DIO チャンネルとリンクされています。**DIO Channel** プルダウンメニューでチャンネル 1 を選択します。入力チャンネル 0－9 に対して Set 0 Upper ボックスにチェックを入れてください。設定の変更を適用する為、*Apply* ボタンをクリックしてください。

トリガモデルの開始

Trigger…Initialize 画面で収集シーケンスが開始されます。

例 2

これは、アラームを使った短期モニタリングアプリケーションの一般的な設定シーケンスの例です。このアプリケーションでの必要条件は以下のとおりです。

- 接続入力はチャンネル 0－47 に対して 48 の E 熱電対を使います。
- モニタされる温度は瞬時的なものである為、1 kHz 周波数フィルタが適しています。
- 測定データは ºC の単位が求められています。
- タイムスタンプの報告が求められています。CJC 温度の報告は必要ありません。

- データは DIO チャンネル 0 の上がりエッジにトリガし、10 秒間の間 1000 測定／秒で収集される必要があります。
- 十秒分のデータが収集されれば測定は終わりです。。
- アラーム表示には DIO チャンネル 7 が利用されます。いずれかの入力チャンネルが 50 ºC を超えた場合、low に移行する必要があります。

このアプリケーションの条件を満たす為には以下の手順を踏んでください。

スキャンリストの設定

Scan List…Configuration 画面でスキャンリストを設定します。*Enabled* リンクをクリックする事によって全 48 チャンネルを適用します。すべてのチャンネルで同じ工学単位変換を行う為、*E* リンクもクリックします。設定の変更を適用する為、*Apply* ボタンをクリックしてください。

フィルタ周波数の設定

Scan List…Filters 画面でフィルタを設定します。全チャンネルで同じフィルタ設定を利用する為、*1 kHz* リンクをクリックします。設定の変更を適用する為、*Apply* ボタンをクリックしてください。

FIFO の設定

Data…FIFO Configuration 画面で FIFO を設定します。**Temperature Units** で°C を選択します。**Report CJC Temperatures** ボックスはアンチェックしてください。**Report Timestamps** ボックスにはチェックを入れます。設定の変更を適用する為、*Apply* ボタンをクリックしてください。

トリガモデルの設定

Trigger…Configuration 画面でトリガモデルを設定します。まずは *Default* ボタンをクリックします。Arm 枝を展開し、Arm Source の設定を表示します。ソースが選択されている場合は全部アンチェックしてください。また、VTI Trig. Bus 枝を展開し、VTB Channel 0 の Pos. Edge を選択します。Trig Source では Timer を選択してください。**TRIG Count** で 10000 と入力します。**Timer Interval** は 0.001 と入力します。後は **Init Continuous** ボックスにチェックを入れます。設定の変更を適用する為、*Apply* ボタンをクリックしてください。

限界値設定

Limits…Limits 画面で限界値条件を設定します。プルダウンメニュー**Set** では Set 0 を選択します。チャンネル 0−47 に対しては **Manual** ボックスにチェックを入れ、上限値として 50 と入力します。設定の変更を適用する為、*Apply* ボタンをクリックしてください。これら限界値条件は **Limits…DIO Limit Events** 画面を通して DIO チャンネルとリンクされています。**DIO Channel** プルダウンメニューでチャンネル 7 を選択します。全入力チャンネルに対して限界値評価をする必要があるため、Set 0 Upper の *All* リンクをクリックしてください。**Invert** ボックスにもチェックを入れます。**Latch** チェックボックスの方ははアンチェックします。設定の変更を適用する為、*Apply* ボタンをクリックしてください。

トリガモデルの開始

Trigger…Initialize 画面で収集シーケンスが開始されます。

第 6 章

プログラミング

はじめに

EX1048 のドライバは上位 C 言語関数で構成された VXI*plug&play* ドライバです。

初期設定

EX1048 のリセット、又は電源の再投入すると機器の設定は初期設定値に戻ります（表 6-1）。 多くのプログラム作業では初期設定値からのパラメータ変更は必要ないので、余剰なコマンドを除くことによってずっと簡単になります。また、 **vtex1048_reset**コマンドを使う事によってEX1048 をいつでもリセット状態に戻すことができます。

トリガ設定リセット値	
設定パラメータ	**リセット値**
Init Continuous	Disabled
Timer Interval (seconds)	0.1
Arm Source	Immediate
Arm Count	1
Arm Delay (seconds)	0
Trig Source	Timer
Trig Count	1
Trig Delay (seconds)	0
スキャンリスト設定リセット値	
設定パラメータ	**リセット値**
Scan List (enabled channels)	0
EU Conversions	mV (voltage)
Filter (hertz)	4.0
FIFO 設定リセット値	
設定パラメータ	**リセット値**
CJC Temperature Reporting	Disabled
Timestamp Reporting	Disabled
Temperature Units	°C
Blocking Mode	Disabled
デジタル I/O 設定リセット値	
設定パラメータ	**リセット値**
Output State	0
Output Enable	Disabled
トリガバス設定リセット値	
設定パラメータ	**リセット値**
Output State	0
Output Enable	Disabled

表6-1: 初期設定

限界値設定 リセット値	
設定パラメータ	**リセット値**
Lower Limit of Limit Set 0	-0.066
Upper Limit of Limit Set 0	0.066
Limit Set 0, Manual	Disabled
Lower Limit of Limit Set 1	-1e+38
Upper Limit of Limit Set 1	1e+38
DIO Limit Events	Disabled
DIO Limit Events, Invert	Disabled
DIO Limit Events, Latch	Disabled
デバイス設定リセット値	
設定パラメータ	**リセット値**
Current Self Cal Data	Cleared
Network Configuration	Unaffected
Time Configuration	Unaffected
Lock	Unaffected
詳細設定リセット値	
設定パラメータ	**リセット値**
User-defined Conversion Coefficients	0
User-defined CJC Temperatures (values)	0
User-defined CJC Temperatures (enable)	Disabled

表 6-1（つづき）: 初期設定

基本命令・クエリ

これらの命令は機器の基本的な設定と通信に使います。命令文法の詳細は Section 7をご参照ください。

デバイスの初期化

vtex1048_init 命令は、EX1048 との通信に使われ、他の命令を実行する以前に使用する必要があります。この命令のキーパラメータは接続しようとしている EX1048 の IP アドレスです。また、この命令は繋がっている機器の ID の確認、および接続時の機器のリセット作業を含んでいます。この命令に対して EX1048 は接続のセッションハンドルを出力し、機器との以後通信にはこの独自なセッションハンドルが必要となります。この為、同じアプリケーション内で、個別のセッションハンドルを使った複数の機器の接続が可能になります。

vtex1048_revisionQuery クエリは、接続している EX1048 のドライバ、及び内蔵ファームウェアのバージョン 情報の取得に使います。

vtex1048_close 命令は、EX1048 との接続の切断に使え、テストアプリケーションの終了時に実行させます。また、この命令を実行させると機器はアンロックされ、次のテストが実行できるようになります。

デバイスのリセット

vtex1048_reset命令は、EX1048 のデータ収集設定パラメータをすべて初期設定値に戻します。一般的に、新しいテストシーケンスを始める前に機器を既知の状態に戻す為に使います。実際に影響を受ける各パラメータとそのリセット値は、表 6-1をご参照くださ

い。機器をリセットすると、現在の自己校正データが消去され、不揮発性メモリ上に自己校正データが存在する場合は、それをロードします。このデータの作成、利用に関する詳しい情報は、この章の **自己校正の実行**画面の説明をご参照ください。

ひとくちメモ 機器のリセットは FIFO リーディングメモリを消去します。収集したデータは、この命令を実行する前に FIFO より回収する必要があります。

自己校正の実行

Section 3の *自己校正*で説明した様に、EX1048 の測定性能は定期的な自己校正をすることによって、大幅に高めることができます。自己校正作業は、短時間で終わる上、入力端子を切断したり外部機器を利用する必要がないので、簡単に何度でも行えます。この自己校正の機能を十分に引き出すには、自己校正定数がどうやって生成、使用、保存されるかを知っておく必要があります。

自己校正によって、完全校正時に生成された不揮発性メモリ内の校正定数が上書きされたり、変更や、置き換えられることはありません。代わりに、測定値の計算時に完全校正定数が適用された後、これを補足する付加的な校正定数が生成されます。自己校正が実行されると、ここで計算されたデータは揮発性メモリ内に置かれます。このデータは “current self cal data”（現在自己校正データ）と呼ばれ、機器によって自動的に使われます。使用する為に命令を実行する必要はありません。ただ、揮発性メモリ内に置かれている為、機器をリセットしたり、又は電源を再投入すると、消去されます。また、“current self cal data”を不揮発性メモリ内に保存することもできます。

自己校正を実行するには、まず機器のロックを取得する必要があります。ロックに関する詳しい情報は、この章の*ロックの取得*の項をご参照ください。ロックが取得されていない状態で自己校正を試みると、エラーが発生します。自己校正処理は **vtex1048_self_cal_init** 命令で実行できます。いったん自己校正が始まってから現在の進行状況を確認するときは、**vtex1048_self_cal_get_status** クエリを利用してください。このパーセント値が 100%になると、自己校正は完了です。

機器のウォームアップ中にユーザーが誤って自己校正を実行することを防ぐ為、EX1048 が 60 分間継続してオンになっていなかった場合は、エラーが発生し、自己校正は実行されません。これは単なる警告であり、無視することもできます。機器が完全にウォームアップする前に自己校正を実行すると、測定性能を下げる可能性があるので、一般的にはあまりお勧めできません。ただ、機器が既にウォームアップされ、一時的な電源の再投入、又は再起動をしただけの場合は、警告を無視してくださっても結構です。この警告無視のメカニズムに関する詳しい情報は Section 7の**vtex1048_self_cal_init**の項をご参照ください。

ひとくちメモ 自己校正前のウォームアップ時間の検出は機器の測定性能を保つ為のものであり、警告を無視する場合は十分に注意する必要があります。ソフトウェアアプリケーションでこの機能を実装する場合は、警告無視が意図的であることを補償する為、ユーザーの関与を必要とする様にすることを強くお勧めします。

上記で説明されたように、自己校正処理を実行すると計算されたデータは揮発性メモリ内に“current self cal data”として記憶されます。このデータは **vtex1048_self_cal_clear** の命令で消去できます。この操作で、揮発性メモリ内のデータは消去されますが、不揮発性メモリ内の自己校正データには影響しません。

自己校正はいつでも実行できますが、自己校正値を不揮発性メモリに保存する必要がある場合もあります。**vtex1048_self_cal_store** の命令は current self cal data を不揮発性メモリ内に保存し、機器の電源再投入、また再起動と共にロードすることができます。current self cal data が存在しない状態でこの命令が実行された場合は、エラーが発生します。不揮発性自己校正データの保存は出荷時の校正設定からの永久的な（取り消しは可能）変更をもたらす行為である為、 その存在を **vtex1048_self_cal_is_stored** クエリで確認することができます。

以前に保存された不揮発性自己校正データは **vtex1048_self_cal_clear_stored** 命令により消去できます。この作業は、不揮発性メモリ内の自己校正データだけを消去する物であり、current self cal data には影響しません。不揮発性自己校正データが存在しない状態でこの命令が実行された場合は、エラーが発生します。逆に、以前に保存された不揮発性自己校正データは **vtex1048_self_cal_load** の命令で current self cal data としてロードすることができます。既に current self cal data が存在する場合は、上書きされますので、事前に消去する必要はありません。不揮発性自己校正データが存在しない状態でこの命令が実行された場合は、エラーが発生します。

current self cal data、及び nonvolatile self cal data（不揮発性自己校正データ）の存在は、自己校正の実行には何の影響もありません。自己校正処理を実行すると、既にあった current self cal data は上書きされます。新しく自己校正をする前に **vtex1048_self_cal_clear** 命令を使う必要はありません。同じく、以前に保存された不揮発性自己校正データの存在は、current self cal data の不揮発性メモリへの保存には影響ありません。以前にあったデータは上書きされますので、事前に消去する必要はありません。

ひとくちメモ 自己校正プロセスの一環として、EX1048 はいったんリセット作業を行います。この時機器の設定パラメータはすべて初期設定値に戻されます。

ロックの取得

初期設定では、EX1048 はウェブ及び機器ドライバの両方のインターフェイスで、複数のホストから無制限のアクセスが可能です。これはユーザーに高い柔軟性をもたらしますが、プロテクトされた作業が望ましい、または必要な場合もあります。そのような場合には、EX1048 を“ロック”することによって **vtex1048_lock** 命令を実行した IP ホスト以外からの命令を受け付けなくすることができます。また、自己校正を実行する場合は、事前にロックを取得する必要があります。

いったんロックが取得されたら、**vtex1048_unlock** 命令で解除することできます。ロック状態は **vtex1048_reset** 命令には影響されないので、これを使ってロックを解除することはできません。

このロックメカニズムは、設計上、他ホストから **vtex1048_break_lock** 命令を実行して解除することが可能です。この為、ロック機能は他ユーザーに対して、プロテクトされた作業が実行中であることを知らせることは出来ますが、安全を保障する物ではありません。これにより、ロックを実行した IP アドレスが利用不能になった場合でも、機器のリカバリーが可能になっています。ロックを解除してもロックは自動的には取得されません。ロックを取得する必要がある場合は、別に **vtex1048_lock** 命令を実行してください。

機器のロック状況は、**vtex1048_check_lock** クエリで確認できます。このクエリによって、ユニットのロックの状態を確認するだけでなく、クエリの発行人がロックの取得者であるか否かも確認できます。

プログラミング・シーケンス

以下のフローチャートは、EX1048 の収集シーケンスをプログラムする際の一般的な手順を表しています。フローチャートを見ると命令に順序関係があるように見えますが、実際には各命令はそれぞれ独立して動作します。ただ、このフローチャートは、作業の論理的な流れを表しています。多くのアプリケーションでは、初期設定値で十分、または特定の機能が必要でない場合など、ここに出てくる全ての操作は必要ありません。

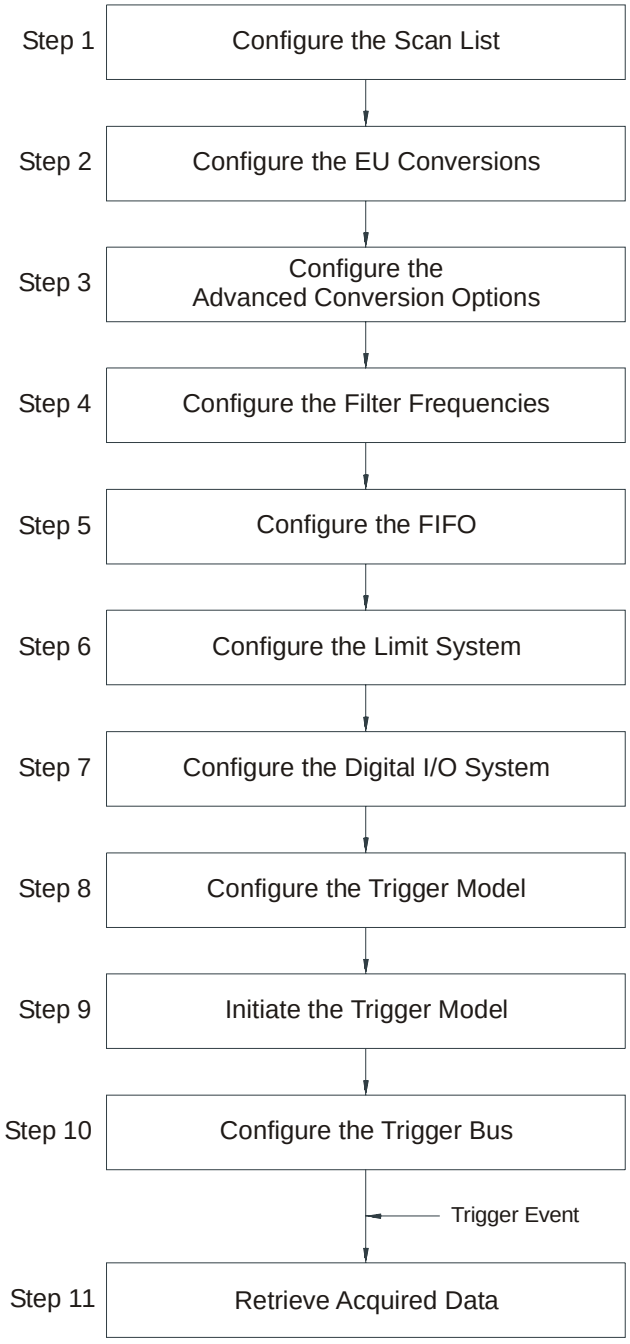


図6-1: プログラミング・シーケンス図

測定命令・クエリ

これらの命令は、データ収集システムの設定、及び収集データの回収に使われます。全命令文法の詳細は Section 7をご参照ください。

スキャンリストの設定

スキャンリストは、**vtex1048_set_scanlist** の命令で定義されます。EX1048 は、1 から全 48 までの入力チャンネルをスキャンリストに設定することができます。スキャンリストは、以下の条件を満たしている必要があります。

- 最低 1 チャンネルを含む。
- 48 チャンネル以下。
- チャンネルの重複がない。

スキャンリスト上のチャンネルは、どのような順番でも構わず、リストされた順番通りにスキャンされます。

例 1: このコードブロックは 0～4 の 5 チャンネルを番号の小さい順にスキャンするよう設定します。

```
ViInt32 channels[5] = { 0, 1, 2, 3, 4 };
vtex1048_set_scanlist(vi, channels, 5);
```

例 2: このコードブロックは 0～4 の 5 チャンネルを番号の大きい順にスキャンするよう設定します。

```
ViInt32 channels[5] = { 4, 3, 2, 1, 0 };
vtex1048_set_scanlist(vi, channels, 5);
```

現在のスキャンリストは、**vtex1048_get_scanlist** クエリで確認できます。

工学単位変換係数の設定

各チャンネルの工学単位変換係数は、**vtex1048_set_channel_conversion** 命令で設定できます。工学単位変換係数は、チャンネルのスキャンリスト上の有無に関わらず設定でき、1 つの命令で複数のチャンネルに同じ工学単位変換係数を割り当てることができます。その代わり、変換タイプの数だけ個別の命令を出す必要があります。

例: このコードブロックでは、チャンネル 0 ～ 4 を E 熱電対に、チャンネル 5 ～ 8 を T 熱電対に設定します。

```
#define TYPE_E    0x04
#define TYPE_T    0x03
ViInt32 e_channels[5] = { 0, 1, 2, 3, 4 };
vtex1048_set_channel_conversion(vi, e_channels, 5, TYPE_E);
ViInt32 t_channels[4] = { 5, 6, 7, 8 };
vtex1048_set_channel_conversion(vi, t_channels, 4, TYPE_T);
```

チャンネルの現在の工学単位変換選択は、**vtex1048_get_channel_conversion** クエリで確認できます。

高度変換設定

利用度を高める為、EX1048 は高レベルなハードウェアとソフトウェアの統合性を提供します。高性能内部冷接点メカニズムによって熱電対線を直接接続することを可能にし、

標準的な熱電対の多項式係数を内蔵することによって外部での計算の必要なく保証された温度を測定することができます。但し、高度設定オプションを利用することにより必要に応じて外部冷接点やユーザー定義の熱電対の計算を行うこともできます。

外部冷接点の使用

EX1048 では、また、ユーザーによって管理・測定された外部冷接点を使用することも可能です。この場合、冷接点温度は ºC で入力され、チャンネル毎に設定の無効・有効を設定します。つまり、内蔵と、ユーザー定義の CJC（冷接点補償）入力を併用することが可能だということです。ユーザー定義 CJC 温度値は **vtex1048_set_user_cjc_temp** の命令で入力できます。CJC 温度値は、チャンネルのスキャンリスト上の有無に関わらず設定でき、1 つの命令で複数のチャンネルに同じ温度値を割り当てることができます。その代わり、温度値の数だけ個別の命令を出す必要があります。

ユーザー定義の CJC 温度値は、**vtex1048_set_user_cjc_enable** 命令でチャンネル毎に有効にします。この場合、熱電対計算は内部測定された物ではなく、ユーザー定義の CJC 温度値を使って実行されます。また、外部 CJC 値の入力とその適用は個別のものであることを覚えておいてください。つまり、値の入力は、その使用を意味する訳ではなく、既に適用された値を無効にしても、その値が消去される訳ではありません。

ひとくちメモ これら命令は外部の熱電対冷接点が利用されたときのみ使ってください。入力チャンネルデータには外部 CJC 値が使用されたことを示す記述は無い為、この機能を使うときは十分に気をつけてください。

チャンネルの現在のユーザー定義 CJC 温度値とその適用状況は、それぞれ **vtex1048_get_user_cjc_temp** と **vtex1048_get_user_cjc_enable** クエリで確認できます。

ユーザー熱電対変換公式の使用

EX1048 は標準的な熱電対に対応し、NIST ITS-90 熱電対データベースの全次数多項式、また係数を使って熱電対計算をします。しかし、アプリケーションによっては、機器内の係数でなくユーザーの定義した係数を使う必要がある場合もあります。この理由は幾つかあります。そのひとつは、利用される熱電対の伝達関数が標準的な熱電対の誤差範囲より高い精度で得られた場合。もうひとつは、標準的でない熱電対を利用する場合です。係数は最高二組が入力可能です。つまり、ユーザー定義熱電対式を利用するにはユーザーが 2 つの変換公式を定義する必要があります。

*forward conversion polynomial（前方変換式）*は、CJC 温度を補償冷接点電圧に変換するときに使われ、次のように記されます。

$$E = c_0 + c_1 * t^1 + c_2 * t^2 + \ldots + c_{12} * t^{12}$$

ここでは、*E* はボルト、*t* は ºC の単位、そして c_0 - c_{12} が係数です。

*inverse conversion polynomial（逆変換式）*は、補償入力電圧を温度に変換するときに使われ、次のように記されます。

$$t = d_0 + d_1 * E^1 + d_2 * E^2 + \ldots + d_{12} * E^{12}$$

ここでは、*E* はボルト、*t* は ºC の単位、そして d_0 - d_{12} が係数です。

これらの係数は、**vtex1048_set_user_conversion** 命令で入力できます。この命令は次のパラメータを受け付けます。

- 工学単位変換タイプ（User0 と User1）にそれぞれ対応する整数値、9 又は 10。入力される係数は、これらの値に割り当てられます。
- 前方変換式の係数の配列。
- 前方変換係数配列の長さに対応する整数値。
- 逆変換式の係数の配列。
- 逆変換係数配列の長さに対応する整数値。

特に定義されていない係数は、自動的に 0 の値にセットされます。各係数の現在の値は、**vtex1048_get_user_conversion** クエリで確認できます。

ひとくちメモ ユーザー定義の係数は、入力しただけでは測定に使用されません。使用するには工学単位変換を User0 か User1 に設定してください。

フィルタ周波数の設定

各チャンネルのハードウェアフィルタは、**vtex1048_set_filt_freq** の命令で設定できます。ハードウェアフィルタは、チャンネルのスキャンリスト上の有無に関わらず設定でき、1 つの命令で複数のチャンネルに特定のフィルタ周波数を割り当てることができます。その代わり、フィルタ周波数の数だけ個別の命令を出す必要があります。

例: このコードブロックでは、チャンネル 0 ~ 4 を 4 Hz フィルタに、チャンネル 5 ~ 8 を 1 kHz フィルタに設定します。

```
ViInt32 low_channels[5] = { 0, 1, 2, 3, 4 };
vtex1048_set_filt_freq(vi, low_channels, 5, 4.0);
ViInt32 high_channels[4] = { 5, 6, 7, 8 };
vtex1048_set_filt_freq(vi, high_channels, 4, 1000.0);
```

チャンネルの現在のハードウェアフィルタの設定値は、**vtex1048_get_filt_freq** クエリで確認できます。

FIFO の設定

vtex1048_set_fifo_config の命令を使って、FIFO メモリのデータ形式とオーバーフロー時の対応の仕方を設定することができます。初期設定では、データ読み出し時に得られる情報はチャンネル読み取り情報とスキャン開始の時刻に限られます。CJC 温度値は、スキャン時には必ず測定されますが、適用されていない限りユーザーに返されることはありません。CJC 温度値は、必ず ºC の単位で報告されます。また、戻り値にチャンネル毎のデルタタイムスタンプを含めることができます。厳密には、このタイムスタンプは、100 ns 単位でチャンネルのスキャン開始から測定までの時間間隔を表しています。

初期設定では、チャンネルの測定は ºC の単位で返されます。この単位は、ユーザーによって ºF に設定することもできます。この設定は、報告される CJC データや工学単位を mV に設定したときの入力チャンネルデータには影響しません。

また、データ形式のほかに、この命令では、スキャン時にリーディングバッファがいっぱいになったときのシステムの対応の仕方を選ぶことができます。ブロッキングモードが適用されている場合は、それ以降の測定データは放棄されます。ブロッキングモードが適用されていない場合は、新しいデータはバッファ内のデータを古い順から上書きしていきます。

機器の現在の FIFO 設定値は、**vtex1048_get_fifo_config** クエリで確認できます。

ひとくちメモ **vtex1048_read_fifo** 命令では、CJC 温度やデルタタイムスタンプ情報は、これらの適用状態に関わらず、入手することはできません。これらの情報は、ストリーミングインターフェイスから入手可能であり、パラメータ設定はここで適用されます。

限界値システムの設定

EX1048 では、二対のプログラマブル限界値を設定することができます。これらの限界値、リミットセット 0 とリミットセット 1 は、チャンネル毎にプログラムが可能です。操作方法はリミットセット間で多少の差があり、リミットセット 0 の方は機能が拡張されています。

リミットセット 0 の値は、初期設定で、各チャンネルの工学単位変換タイプと温度単位設定を元に自動的に決められます。つまり、リミットセット 0 の上限値と下限値は、表 3-2で表記されているEX1048 の測定範囲の上限値と下限値に設定されます。例えば、入力チャンネル 0 がE熱電対とºCの単位に設定されている場合、上限値と下限値はそれぞれ+900 と-200 となります。この自動設定を無効にし、チャンネル毎に手動で上限値と下限値を入力するには、まず**vtex1048_set_limit_set0_manual**命令で手動入力設定を有効にする必要があります。手動入力設定は、チャンネルのスキャンリスト上の有無に関わらず設定でき、1 つの命令で複数のチャンネルに同一の値を設定することができます。その代わり、設定値の数だけ個別の命令を出す必要があります。

いったん手動入力モードを有効にすると、ユーザーは **vtex1048_set_limit_set0** 命令でチャンネルのリミットセット 0 を入力することができます。限界値は、チャンネルがスキャンリスト上に在るか否かに関わらず設定でき、1 つの命令で複数のチャンネルに特定の限界値ペアを割り当てることができます。その代わり、限界値の組み合わせの数だけ個別の命令を出す必要があります。

例: このコードブロックでは、チャンネル 0～4 を 0 と 100 の手動限界値に設定します。

```
ViInt32 e_channels[5] = { 0, 1, 2, 3, 4 };
vtex1048_set_limit_set0_manual(vi, e_channels, 5, 1);
vtex1048_set_limit_set0(vi, e_channels, 5, 0, 100);
```

逆に、チャンネルの手動限界値設定を無効にすると、チャンネルのリミットセット 0 の値は現在の工学単位変換タイプと温度単位設定を元に自動的に決められます。

チャンネルの現在の手動入力設定とリミットセットの 0 値は、それぞれ **vtex1048_get_limit_set0_manual** と **vtex1048_get_limit_set0** クエリで参照できます。

リミットセット 1 の方は、手動モードでのみ働きます。チャンネルのリミットセット 1 の値は、**vtex1048_set_limit_set1** 命令で設定します。限界値は、チャンネルがスキャンリスト上に在るか否かに関わらず設定でき、1 つの命令で複数のチャンネルに特定の限界値ペアを割り当てることができます。その代わり、限界値の組み合わせの数だけ個別の命令を出す必要があります。

チャンネルの現在のリミットセット 1 の値は、**vtex1048_get_limit_set0** クエリで参照できます。

ひとくちメモ マニュアルモードで動作しているリミットセットは工学単位変換タイプや温度単位設定の変更に合わせて自動的に更新されることはありません。間違えの無い様、限界値は工

学単位変換タイプや温度単位の設定が終わってから行ってください。また、設定が変更された場合は限界値も忘れずに変更してください。

収集が終わると、**vtex1048_get_accum_limit_status** クエリで全 48 チャンネル分の累積した限界値状況を得ることができます。このクエリに対する応答は各 48 の論理値を含む 4 つの配列です。この 4 つの配列はそれぞれ set 0 下限値、set 0 上限値、set 1 下限値、set 1 上限値を表しています。これら値は、前回のトリガの開始からの測定データの各限界値状況をチャンネル毎に表します。つまり、“1” の返答値は限界値の超過、 “0” は限界値に達していないことを示しています。限界値の状態は **vtex1048_init_imm** の一環として消去されます。限界値状態は、チャンネルがスキャンリスト上に在るか否かに関係なく全チャンネル分が返されます。スキャンされていないチャンネルの限界値状態は常に “0” です。

デジタル I/O システムの設定

EX1048 のデジタル I/O ポートは、入出力の作業に使用することができます。各 DIO チャンネルは出力機能、出力時のスタティックレベル、パルス動作において個別にプログラム可能です。

vtex1048_get_dio_input クエリは、DIO チャンネルで測定されたデジタルレベル（0 又は 1）を取得します。入力モニタリング機能はチャンネルが出力として同時使用されている場合でも機能します。このクエリに対する返答はポートの 8 ビット値を表す 0 から 255 の十進数で表わされます。この 8 ビットフィールドの内、MSB は DIO チャンネル 7、LSB は DIO チャンネル 0 に対応します。

例: このコードブロックでは、現在の I/O ポートの状態をクエリし、ビット 7、4、0 のレベルをレポートします。

```
ViInt32 dio_in;
vtex1048_get_dio_input(vi, &dio_in);
if (dio_in & 0x80)
    printf(“Bit 7 is high”);
else printf(“Bit 7 is low”);
if (dio_in & 0x10)
    printf(“Bit 4 is high”);
else printf(“Bit 4 is low”);
if (dio_in & 0x01)
    printf(“Bit 0 is high”);
else printf(“Bit 0 is low”);
```

vtex1048_set_dio_output の命令は、出力として適用されたときデジタル I/O ポートの各チャンネルが取るスタティックレベルを設定します。出力の有効／無効は **vtex1048_set_dio_output_enable** の命令で行います（次段参照）。この命令は、8 ビットポートの求められた状態を示す十進数（0 - 255）または十六進数（0x00 - 0xFF）の値を受け付けます。この 8 ビットフィールドの内、MSB は DIO チャンネル 7、LSB は DIO チャンネル 0 に対応します。

vtex1048_set_dio_output_enable 命令は、デジタル I/O ポートの各チャンネルの出力機能の有効／無効を設定します。いったん有効になると、チャンネルは **vtex1048_set_dio_output** 命令で設定したレベルになります（前段参照）。出力として適用されていない場合、チャンネルは 3 ステート状態となります。各チャンネルの入力状態は出力状態に関係なく動作します。この命令は、8 ビットポートの出力の有効状態を示す十進数（0 - 255）または十六進数（0x00 - 0xFF）の値を受け付けます。この 8 ビットフィールドの内、MSB は DIO チャンネル 7、LSB は DIO チャンネル 0 に対応します。

例: このコードブロックは I/O ポートのビット 7（high）とビット 6（low）を選択し、出力として適用します。

```
vtex1048_set_dio_output(vi, 0x80);
vtex1048_set_dio_output_enable(vi, 0xC0);
```

vtex1048_get_dio_output クエリは、プログラムされた DIO チャンネルの出力ステートを求めるときに使われます。このクエリに対する返答は、8 ビットポートのプログラムされた出力ステートを示す 0 から 255 の十進数です。但し、出力は有効にされる必要があるため、これは必ずしも DIO チャンネルの実際の出力ステートをあらわすものではありません。**vtex1048_get_dio_output_enable** クエリは、各チャンネルの出力有効状態を調べるときに使われます。このクエリに対する返答は、8 ビットポートの有効状態を示す 0 から 255 の十進数です。この 8 ビットフィールドの内、MSB は DIO チャンネル 7、LSB は DIO チャンネル 0 に対応します。

例: このコードブロックは、I/O ポートの出力ステート、出力イネーブルステートをクエリし、ビット 4 のステートをレポートします。

```
ViInt32 dio_out;
ViInt32 dio_outen;
vtex1048_get_dio_output(vi, &dio_out);
vtex1048_get_dio_output_enable(vi, &dio_outen);
if (dio_out & 0x10)
      printf("Bit 4 is set high");
else printf("Bit 4 is set low");
if (dio_outen & 0x10)
      printf("Bit 4 is enabled");
else printf("Bit 4 is not enabled");
```

vtex1048_set_dio_pulse 命令は選択されたデジタル I/O ポートのチャンネルから 1 µs のパルスを出力するときに使われます。パルスは選択されたチャンネルが出力として適用されている場合にのみ発生されます。チャンネルのスタティックレベルが high にプログラムされている場合、パルスは low になります。また、チャンネルのスタティックレベルが low にプログラムされている場合は、パルスは high になります。各パルスの出力毎に個別に命令を実行する必要があります。この命令は、8 ビットポートの内、パルスが出力されるチャンネルを示す十進数（0 - 255）または十六進数（0x00 - 0xFF）の値を受け付けます。この 8 ビットフィールドの内、MSB は DIO チャンネル 7、LSB は DIO チャンネル 0 に対応します。

例: このコードブロックではビット 7 をスタティックレベル low に設定し、その後 high パルスを出力します。

```
vtex1048_set_dio_output(vi, 0x00);
vtex1048_set_dio_output_enable(vi, 0x80);
vtex1048_set_dio_pulse(vi, 0x80);
```

DIO リミットイベントの設定

簡単な入出力の作業のほかにも、デジタル I/O ポートは入力チャンネルの限界値状態に関連付けることができます。この関連付けは DIO リミットイベントと呼ばれます。この DIO リミットイベントはチャンネル毎にプログラム可能です。つまり、各 8 つの DIO チャンネルはそれぞれ個別の動作性質を持たせることが出来ます。初期設定では、入力チ

ャンネルの限界値評価に関連付けされた DIO チャンネルは、限界値を超えると low から high に移行します。

DIO チャンネルを特定の入力チャンネルの特定の限界値条件に結びつける作業は **vtex1048_set_dio_limit_event** 命令によってなされます。複数個の選択も可能で、論理的には OR で結び付けられます。つまり、4 つの限界値条件に対応付けられた DIO チャンネルは指定された 4 つの限界値のいずれかひとつが満たされれば状態を移行することになります。EX1048 では同じ入力チャンネル、及び複数の入力チャンネルをまたいで幾つもの結び付けを決めることが可能です。

この命令は以下のパラメータを受け付けます。

- DIO チャンネル番号（0 – 7）を示す整数値。
- 入力チャンネル単位で 4 つの限界値条件の評価との関連性を表す 48 個の 4 ビット整数値の配列。この 4 ビットフィールド内では、リミットセット 0 下限、リミットセット 0 上限、リミットセット 1 下限、リミットセット 1 上限、の順序で値が並べられています。この値は十進数（0 – 15）または十六進数（0x00 – 0x0F）で表すことが出来ます。チャンネル 0 からチャンネル 47 まではそれぞれ配列の要素[0]から[47]で表されています。

例: このコードブロックは、DIO チャンネル 7 を入力チャンネル 12 と 14 でリミットセット 1 下限、そして入力チャンネル 1 でリミットセット 0 上限に関連付けします。

```
ViUInt8 limit_masks[48] =
{0,4,0,0,0,0,0,0,0,0,0,0,2,0,2,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0};
vtex1048_set_dio_limit_event(vi, 7, limit_masks);
```

DIO リミットイベントとスキャンリストの設定は個別に行われる作業です。つまり、限界値条件を特定の入力チャンネルと関連付けするだけではスキャンリストに適用されることはありません。同じく、入力チャンネルをスキャンリストから削除しても自動的に限界値条件との関連付けがなくなることはありません。スキャンリスト上にないチャンネルへ関連付けられた DIO リミットイベントは限界値評価の時には無視されるので、誤動作の原因となることはありません。

リミットイベントとして関連付けされた場合、DIO チャンネルは新しい収集の開始に初期化されます。そして、そのステートはプログラムされた限界値の評価に応じてスキャン毎に更新されます。初期設定では、DIO の初期化された状態は low ですが、**vtex1048_set_dio_limit_event_invert** の命令によってチャンネルごとに high にセットすることができます。同様に、各チャンネルの初期動作設定は非ラッチモードですが、**vtex1048_set_dio_limit_event_latch** の命令によってチャンネルごとにラッチモードにすることができます。ラッチモードでは、将来の限界値の評価結果に関係なく、新しい収集シーケンスで初期化されるまで、初期化状態からの移行は保持されます。

DIO リミットイベントメカニズムの現在状況は **vtex1048_get_dio_limit_event**、**vtex1048_get_dio_limit_event_invert**、**vtex1048_get_dio_limit_event_latch** のクエリで取得することができます。

トリガソースの設定

EX1048 では、独自のアームソースとトリガソースイベント構造を備えたフルファンクショントリガモデルをサポートしています。トリガモデルに関する詳しい説明は Section 4をご参照ください。要約すると、収集シーケンスは、トリガ開始命令によって適用されます。この後、プログラムされたアームソースイベントの受け取り、続くトリ

ガソースイベントの受け取りと共に、スキャンが始まります。トリガ、及びアームソースイベントは、Immediate（即時）、Timer（タイマー）、Digital I/O（デジタルI/O）、Trigger Bus（トリガバス）など無数のソースより個別にプログラム可能です。

ARM イベントシステムの設定

ARM ソースは **vtex1048_set_arm_source** の命令で設定できます。ARM イベントはトリガバスチャンネルエッジ／レベル、デジタル I/O チャンネルエッジ／レベル、タイマー時間間隔、の任意の組み合わせや、または即時動作に設定することができます。複数のARM イベントソースが指定される場合、次のように論理的に組み合わされます。

ARM event = [(Timer tick event) AND (Digital I/O event) AND (Trigger Bus event)]

各デジタルハードウェアポート内でもまた特定のチャンネルに対して複数のチャンネルと複数の条件を指定することができます。その場合、次のように論理的に組み合わされます。

Digital I/O event = (Ch 7 events) AND (Ch 6 events) ... AND (Ch 0 events)

最後に、各チャンネル内でのよっつのイベント条件、Pos. Edge, Neg. Edge, Pos. Level, Neg. Level（上がりエッジ、下りエッジ、正相レベル、負相レベル）は OR で組み合わされます。

例えば、デジタル I/O イベントがチャンネル 3 の正相レベル、チャンネル 6 の上がりエッジ、そしてチャンネル 6 の下りエッジの組み合わせと定義された場合、チャンネル 3 の正相レベルがチャンネル 6 の上がりか下りのエッジと同時に起こった時にイベントが満たされることになります。ここではデジタル I/O イベント構成が例として出されましたが、トリガバスイベント構成も同じように働きます。

この命令は以下のパラメータを受け付けます。

- トリガバスの 8 チャンネル上のイベント適用を示す 4 つの 8 ビット値の配列。値は、上がりエッジ、下りエッジ、正相レベル、負相レベルの順になっています。各値は十進数（0 - 255）または十六進数（0x00 - 0xFF）で表されます。この 8 ビットフィールドの内、MSB は VTB チャンネル 7、LSB は VTB チャンネル 0 に対応します。
- デジタル I/O ポートの 8 チャンネル上のイベント適用を示す 4 つの 8 ビット値の配列。値は、上がりエッジ、下りエッジ、正相レベル、負相レベルの順になっています。各値は十進数（0 - 255）または十六進数（0x00 - 0xFF）で表されます。この 8 ビットフィールドの内、MSB は DIO チャンネル 7、LSB は DIO チャンネル 0 に対応します。
- Timer（タイマー）イベントの適用状態を示す論理値。
- Immediate（即座動作）イベントの適用状態を示す論理値。

ソフトウェアアームはここの設定に関係なく常に適用されています。

例 1: このコードブロックは、アームをタイマーのみに設定します。

```
ViUInt8 vtb_masks[4] = {0,0,0,0};
ViUInt8 dio_masks[4] = {0,0,0,0};
vtex1048_set_arm_source(vi, vtb_masks, dio_masks, 1, 0);
```

例 2: このコードブロックでは、DIO チャンネル 0 ~ 3 は正相レベルに、VTB チャンネル 6 は立下り時にアームします。

```
ViUInt8 vtb_masks[4] = {0,64,0,0};
ViUInt8 dio_masks[4] = {0,0,0x0F,0};
vtex1048_set_arm_source(vi, vtb_masks, dio_masks, 0, 0);
```

例 3: このコードブロックは、アームをソフトウェアアームのみに設定します。

```
ViUInt8 vtb_masks[4] = {0,0,0,0};
ViUInt8 dio_masks[4] = {0,0,0,0};
vtex1048_set_arm_source(vi, vtb_masks, dio_masks, 0, 0);
```

現在の ARM ソースの設定は、**vtex1048_get_arm_source** クエリで確認できます。

ARM カウントは **vtex1048_set_arm_count** 命令で 1 から(2^{31}-1)間の数値に設定できます。この値は各トリガ開始時、または init continuous が適用されている場合は 0 に達すると自動的にリセットされます。

例: このコードブロックは ARM カウントを 10 に設定します。

```
vtex1048_set_arm_count(vi, 10);
```

現在の ARM カウントの設定は **vtex1048_get_arm_count** クエリで確認できます。

また、ARM カウントは **vtex1048_set_arm_infinite** 命令によって、ARM カウント値の設定を無視し、ARM カウントを「無限」に設定することができます。この命令は単純に無限 ARM カウントの適用を表す論理値を受け付けます。

現在の無限 ARM カウント設定は **vtex1048_get_arm_infinite** クエリで参照できます。

ARM 遅延は 0 から 4294 秒までを 0.000001 (1 µs)の分解能で、**vtex1048_set_arm_delay** 命令によって指定できます。ARM 遅延は ARM イベント確認時から TRIG レイヤーへの移行までの時間遅れです。

例: このコードブロックは 5 ms の ARM 遅延を設定します。

```
vtex1048_set_arm_delay(vi, 0.005);
```

現在の ARM 遅延は **vtex1048_get_arm_delay** クエリで要求できます。

TRIG イベントシステムの設定

TRIG ソースは **vtex1048_set_trigger_source** 命令で設定できます。TRIG イベントはトリガバスチャンネルエッジ／レベル、デジタル I/O チャンネルエッジ／レベル、タイマー時間間隔、の任意の組み合わせ、または即時動作に設定することができます。複数の TRIG イベントソースが指定される場合、次のように論理的に組み合わされます。

```
TRIG event = [(Timer tick event) AND (Digital I/O event) AND (Trigger Bus event)]
```

各デジタルハードウェアポート内でもまた特定のチャンネルに対して複数のチャンネルと複数の条件を指定することができます。その場合、次のように論理的に組み合わされます。

```
Digital I/O event = (Ch 7 events) AND (Ch 6 events) ... AND (Ch 0 events)
```

最後に、各チャンネル内での 4 つのイベント条件、Pos. Edge, Neg. Edge, Pos. Level, Neg. Level（上がりエッジ、下りエッジ、正相レベル、負相レベル）は OR で組み合わされます。

例えば、デジタル I/O イベントがチャンネル 3 の正相レベル、チャンネル 6 の上がりエッジ、そしてチャンネル 6 の下りエッジの組み合わせと定義された場合、チャンネル 3 の正相レベルがチャンネル 6 の上がりか下りのエッジと同時に起こった時にイベントが満たされることになります。ここではデジタル I/O イベント構成が例として出されましたが、トリガバスイベント構成も同じように働きます。

この命令は以下のパラメータを受け付けます。

- トリガバスの 8 チャンネル上のイベント適用を示す 4 つの 8 ビット値の配列。値は、上がりエッジ、下りエッジ、正相レベル、負相レベルの順になっています。各値は十進数（0 - 255）または十六進数（0x00 - 0xFF）で表されます。この 8 ビットフィールドの内、MSB は VTB チャンネル 7、LSB は VTB チャンネル 0 に対応します。
- デジタル I/O ポートの 8 チャンネル上のイベント適用を示す 4 つの 8 ビット値の配列。値は、上がりエッジ、下りエッジ、正相レベル、負相レベルの順になっています。各値は十進数（0 - 255）または十六進数（0x00 - 0xFF）で表されます。この 8 ビットフィールドの内、MSB は DIO チャンネル 7、LSB は DIO チャンネル 0 に対応します。
- Timer（タイマー）イベントの適用状態を示す論理値。
- Immediate（即座動作）イベントの適用状態を示す論理値。

ソフトウェアトリガはここの設定に関係なく常に適用されています。

例 1: このコードブロックはトリガをタイマーのみに設定します。

```
ViUInt8 vtb_masks[4] = {0,0,0,0};
ViUInt8 dio_masks[4] = {0,0,0,0};
vtex1048_set_trigger_source(vi, vtb_masks, dio_masks, 1, 0);
```

例 2: このコードブロックは DIO チャンネル 0 ~ 3 を正相レベルトリガ、VTB チャンネル 6 を立下りトリガ（エッジトリガ）に設定します。

```
ViUInt8 vtb_masks[4] = {0,64,0,0};
ViUInt8 dio_masks[4] = {0,0,0x0F,0};
vtex1048_set_trigger_source(vi, vtb_masks, dio_masks, 0, 0);
```

例 3: このコードブロックはトリガをソフトウェアトリガのみに設定します。

```
ViUInt8 vtb_masks[4] = {0,0,0,0};
ViUInt8 dio_masks[4] = {0,0,0,0};
vtex1048_set_trigger_source(vi, vtb_masks, dio_masks, 0, 0);
```

現在の TRIG ソースは **vtex1048_get_trigger_source** クエリで参照できます。

TRIG カウントは 1 から$(2^{31}-1)$までの値を **vtex1048_set_trigger_count** の命令で指定できます。この値は各 ARM イベント時にリセットされます。

例: このコードブロックは TRIG カウントを 10 に設定します。

```
vtex1048_set_trigger_count(vi, 10);
```

現在の TRIG カウントは **vtex1048_get_trigger_count** クエリで参照できます。

vtex1048_set_trigger_infinite 命令によって手動 TRIG カウントの設定を無視し、TRIG カウントを「無限」に設定することができます。この命令は単純に無限 TRIG カウントの適用を表す論理値を受け付けます。

現在の無限 TRIG カウント設定は **vtex1048_get_trig_infinite** クエリで参照できます。

TRIG 遅延は 0 から 4294 秒までを 0.000001 (1 µs)の分解能で、**vtex1048_set_trigger_delay** 命令によって指定できます。TRIG 遅延は TRIG イベント確認時からスキャンリストの開始までの時間です。

例: このコードブロックは 5 ms の TRIG 遅延を設定します。

```
vtex1048_set_trigger_delay(vi, 0.005);
```

現在の TRIG 遅延時間は **vtex1048_get_trigger_delay** クエリで参照できます。

基本トリガモデルパラメータの設定

Timer ソースイベントの時間間隔は **vtex1048_set_trigger_timer** 命令によって設定できます。このときの値の範囲は 0.000001 (1 µs)の分解能で 0.001 (1 ms)から 4294 の数値が入ります。この時間間隔は Timer イベントソースの頻度を制御します。アーム、トリガ両イベント同じ値が使われることになります。

例: このコードブロックは 50 ms の時間間隔を設定します。

```
vtex1048_set_trigger_timer(vi, 0.05);
```

現在の時間間隔は **vtex1048_get_trigger_timer** クエリで参照できます。

普段、各収集シーケンスはトリガ開始命令で開始されます。しかし、この必要条件は Init Continuous モードの使用で回避することができます。つまり、Init Continuous は新しくトリガ開始命令を実行しなくともトリガモデルを ARM レイヤーの入り口に戻すのです。これによって特定の収集シーケンスは自動的に再施行させることができます。Init Continuous モードの適用は **vtex1048_set_init_cont** 命令で行うことが出来ます。この命令は単純に Init Continuous モードの適用状態を表す論理値を受け付けます。

現在の Init Continuous 設定は **vtex1048_get_init_cont** クエリで参照できます。

トリガモデル設定のリセット

vtex1048_reset_trigger_arm命令はEX1048 のトリガ設定パラメータをすべて初期設定に戻すときに使います。**vtex1048_reset**命令によく似ていますが、これはトリガ設定パラメータのみに影響します。これらのパラメータとリセット値は 表 6-1をご参照ください。

ひとくちメモ トリガモデルシステムの高い柔軟性は高度に特殊なトリガ条件の作成を可能にすることによって、強力なアプリケーションツールとなり得ます。しかし、これは同時に実際には満たすことの難しいトリガ条件の作成を許すことにもなります。例えば、二つのエッジが同時に起こったと認識するにはお互いの時間差が 25 ns 未満である必要があります。同じように、タイマーソースはデジタルハードウェアエッジ条件と組み合わせるべきではありません。

トリガモデルの施行

トリガの施行は **vtex1048_init_imm** 命令で行います。この命令を受け取ると、トリガモデルは IDLE レイヤーより移行し、ARM イベント待ちとなります。この命令は FIFO リーディングメモリの消去と限界値状態のリセットも行います。

いったんトリガモデルが IDLE レイヤーを出ると、殆どの機器設定操作は無効になります。但し、収集の方は **vtex1048_abort** 命令によっていつでも中止することが出来ます。収集の中止は現在保存されているデータには影響ありません。

トリガバスの設定

トリガバスの基本的な使用目的は複数ユニットのトリガと高速同期信号の伝達です。その為、入力と出力の両方に対応しています。各 VTB チャンネルはその出力機能、出力時のスタティックレベル、そしてパルス動作に対して個別にプログラムが可能であります。

vtex1048_get_vtb_input クエリで VTB チャンネルの測定されたデジタルレベル（0 または 1）を参照できます。入力モニタリング機能はチャンネルの出力状態に関係なく動作します。このクエリに対する返答はバスの 8 ビット値を表す 0 から 255 の十進数です。8 ビットフィールドの内、MSB は VTB チャンネル 7、LSB は VTB チャンネル 0 に対応します。

例: このコードブロックはトリガバスの状態を参照し、ビット 7、4、0 の値を返します。

```
ViInt32 vtb_in;
vtex1048_get_vtb_input(vi, &vtb_in);
if (vtb_in & 0x80)
    printf("Bit 7 is high");
else printf("Bit 7 is low");
if (vtb_in & 0x10)
    printf("Bit 4 is high");
else printf("Bit 4 is low");
if (vtb_in & 0x01)
    printf("Bit 0 is high");
else printf("Bit 0 is low");
```

vtex1048_set_vtb_output 命令は、出力として適用されたときトリガバスの各チャンネルが取るスタティックレベルを設定します。ただ、出力の有効設定の方は **vtex1048_set_vtb_output_enable** 命令で行います（次段参照）。この命令は、8 ビットバスの有効状態を示す十進数（0 - 255）または十六進数（0x00 - 0xFF）の値を受け付けます。この 8 ビットフィールドの内、MSB は VTB チャンネル 7、LSB は VTB チャンネル 0 に対応します。

vtex1048_set_vtb_output_enable 命令は、トリガバスの各チャンネルの出力機能の有効／無効の設定をします。いったん有効になると、チャンネルは **vtex1048_set_vtb_output** 命令で設定したレベルになります（前段参照）。出力として適用されていない場合、チャンネルは 3 ステート状態となります。各チャンネルの入力状態は出力状態に関係なく動作します。この命令は、8 ビットバスの出力の適用状態を示す十進数（0 - 255）または十六進数（0x00 - 0xFF）の値を受け付けます。この 8 ビットフィールドの内、MSB は DIO チャンネル 7、LSB は DIO チャンネル 0 に対応します。

例: このコードブロックはトリガバスのビット 7（high）とビット 6（low）を選択し、出力として適用します。

```
vtex1048_set_vtb_output(vi, 0x80);
vtex1048_set_vtb_output_enable(vi, 0xC0);
```

vtex1048_get_vtb_output クエリは、プログラムされたトリガバスの出力ステートを求めるときに使われます。このクエリに対する返答は、8 ビットバスのプログラムされた出力ステートを示す 0 から 255 の十進数です。但し、出力は有効にされる必要があるため、これは必ずしもトリガバスの実際の出力ステートをあらわすものではありません。**vtex1048_get_vtb_output_enable** クエリは、各チャンネルの出力有効状態を求めるときに使われます。このクエリに対する返答は、8 ビットポートの有効状態を示す 0 から 255 の十進数です。この 8 ビットフィールドの内、MSB は VTB チャンネル 7、LSB は VTB チャンネル 0 に対応します。

例: このコードブロックは、トリガバスの出力ステート、出力有効状態を参照し、ビット 4 のステートをレポートします。

```
ViInt32 vtb_out;
ViInt32 vtb_outen;
vtex1048_get_vtb_output(vi, &vtb_out);
vtex1048_get_vtb_output_enable(vi, &vtb_outen);
if (vtb_out & 0x10)
    printf("Bit 4 is set high");
else printf("Bit 4 is set low");
if (vtb_outen & 0x10)
    printf("Bit 4 is enabled");
else printf("Bit 4 is not enabled");
```

vtex1048_set_vtb_pulse 命令は選択されたトリガバスのチャンネルから 1 µs のパルスを出力するときに使われます。パルスは選択されたチャンネルが出力として適用されている場合のみ出力されます。チャンネルのスタティックレベルが high にプログラムされている場合、パルスは low になります。チャンネルのスタティックレベルが low にプログラムされている場合は、パルスは high になります。各パルス出力毎に、個別に命令を実行する必要があります。この命令は、8 ビットポートの内パルスされるチャンネルを示す十進数（0 - 255）または十六進数（0x00 - 0xFF）の値を受け付けます。この 8 ビットフィールドの内、MSB は VTB チャンネル 7、LSB は VTB チャンネル 0 に対応します。

例: このコードブロックはビット 7 をスタティックレベルを low に設定し、その後 high パルスを出力します。

```
vtex1048_set_vtb_output(vi, 0x00);
vtex1048_set_vtb_output_enable(vi, 0x80);
vtex1048_set_vtb_pulse(vi, 0x80);
```

トリガイベント

収集シーケンスはプログラムされたアーム、及びトリガイベントが認識され次第開始されます。プログラムされたイベントソースに加えて、ソフトウェア・アームとソフトウェア・トリガは常に適用されています。これらはトリガモデルをソフトウェアプログラムで直接制御する時に使われます。

ソフトウェア・アームは **vtex1048_soft_arm** 命令で実行されます。ソフトウェア・アームは、命令が出された時点で適用され、システムが ARM イベント待ちでない場合は無視されます。ソフトウェア・アームの実行は、ARM ソースの設定には影響されません。

ソフトウェア・トリガは **vtex1048_soft_trigger** 命令で実行されます。ソフトウェア・トリガは、命令が出された時点で適用され、システムが TRIG イベント待ちでない場合は

無視されます。ソフトウェア・トリガの実行は、TRIG ソースの設定には影響されません。

収集データの読み出し

EX1048 は収集データを読み出しする前にバッファする為に 20 MBのFIFOメモリを利用します。バッファ可能なスキャン数はスキャンリスト上のチャンネル数とデータフォーマットに依存します。これに関する特定の公式及び使用例は Section 3の *データ収集*の項を参照ください。FIFO メモリ内の未回収データページ（スキャン）数は **vtex1048_get_fifo_count**クエリで参照できます。

EX1048 の収集データは **vtex1048_read_fifo** 命令により読み出しすることが出来ます。この命令のパラメータにより、読み出したいデータスキャン数、及びデータ転送の待ち時間を設定することができます。この命令で取得できるデータは入力チャンネル測定データと各スキャンの開始時刻です。CJC チャンネル測定データや各チャンネルのタイムスタンプデータはここに含まれておりません。これらの情報が必要な場合はストリーミングインターフェイスを使用してください。

この命令の文法は複雑な為、ユーザーが各入力パラメータと戻り値の意味を理解することが重要になります。この命令でもうひとつ大事な概念はブロッキング命令です。つまり、いったんこの命令がアプリケーションによって出された場合、**vtex1048_read_fifo** 命令が実行完了するまでそのアプリケーション内では他の命令を受け付けません。

この実行命令を完了する方法は二通りあります。ひとつは要求されたスキャンがすべて EX1048 によって送り届けられた場合。これは実行の成功とみなされます。もうひとつは指定されたタイムアウト時間が過ぎた場合。この場合は要求された数のスキャンは送り届けられていません。その為、実行エラーとみなされます。但し、アプリケーションによってはこの様な動作が想定され、必要な場合もあります。この場合、厳密にはエラー状態となるのですが、返ってくるデータはどれも問題のないものです。これらの概念は下の項で詳しく説明されています。

この命令は以下の命令パラメータを受け付けます。

- 最大スキャン数（*maxscans*）。これは要求されるスキャン数を表しています。命令実行時に FIFO count ≥ *maxscans* である場合は *maxscans* が返され、命令が完了します。逆に命令実行時に FIFO count < *maxscans* である場合は *maxscans* が返されるかタイムアウト時間に達するまで EX1048 をポールしつづけます（下段参照）。
- スキャン開始時刻を 1970 年 1 月 1 日からの秒数で表した戻り配列（*ts_secs[]*）。データの整合性を保つ為、*ts_secs[]*の宣言された次元は少なくとも *maxscan* 以上である必要があります。
- スキャン開始時刻を *ts_secs[]*の最後の（整数）秒からの経過時間で表した戻り配列（*ts_usecs[]*）。データの整合性を保つ為、*ts_usecs[]*で宣言された配列の大きさは少なくとも *maxscan* 以上である必要があります。
- 実際に回収されたスキャン数を表す戻り値（*numscans*）。命令が *maxscans* に達することによって完了した場合は、*numscans* は *maxscans* と同じです。命令がタイムアウトによって完了した場合は *numscans* は *maxscans* より少ない数値となります。
- 戻りデータ配列の最大長（*maxdata*）。これは *data[]*配列（次段参照）の宣言された大きさです。データの整合性を保つ為、*maxdata* は *maxscans* にスキャンリスト上のチャンネル数をかけた数字と同等かそれ以上である必要があります。FIFO からのデータは *maxdata* パラメータでなく *maxscans* パラメータを元に落とされる為、*maxdata* 値が小さすぎるとデータが永遠に失われる可能性があります。
- サンプルデータの戻り配列（*data[]*）。

- 実際に回収されたサンプル数を表す戻り値（*numdata*）。これは *maxdata* か *numscans* にスキャンリスト上のチャンネルをかけた数字のどちらか少ない方と同じです。
- EX1048 から（*maxscan* を回収する為に）データをポールするタイムアウト時間（*to_secs*）を秒で表した戻り値。 0 の値は無限のタイムアウト時間を表しています。

to_secs パラメータの重要性は使用されるトリガモデル構成に依存します。一般的な ARM が Immediate（即時動作）で TRIG が Timer（タイマー）に設定されているアプリケーションにおいては回収されるデータ量とデータ回収に掛かる時間はそのまま決定されます。つまり、トリガ開始において TRIG カウントに同等であるスキャン数があり、全部で（TRIG カウント） x （タイマー時間間隔）の間、収集されることになります。例えば、500 スキャンを 5 ms 間隔で収集するのに掛かる時間は 2.5 秒です。この場合、*to_secs* は 2.5 より大きな数字であれば何でも良い事になります。この場合、タイムアウトによる命令の完了はエラー状態となります。

これと対照的な例として、特定のテスト条件が整ってからのデータが要求されるケースを考えてみましょう。その為には ARM ソースはデジタル I/O ポートの特定条件に結び付けられているとします。また、このデジタルイベントは EX1048 のデータを受け取っている同じアプリケーションによって非同期的に発生されているとしましょう。そしてこのコントロールアプリケーションのロジックは毎回、前より大きい値の信号を送り、その後 EX1048 に対してデータがポールされます。この場合、もしデータが返されたら EX1048 がトリガされた事になり、それ以上大きな信号は必要なくなります。もし一定時間を経てもデータが返されない場合は信号は更に大きくなります。この場合、タイムアウト時間は信号システムが更新される頻度を決める為、とても重要になります。また、**vtex1048_read_fifo** 命令は最初はタイムアウトによって終了することが想定されている為、ユーザーによる介入を必要とするエラー状態とはみなされません。

例: このコードブロックでは、タイマースキャン後に FIFO を読み込みます。

```
#define NUM_CHANNELS            5
#define MAX_SCANS               20
#define MAX_DATA                (NUM_CHANNELS * MAX_SCANS)
#define TIMER_INTERVAL          0.01

ViInt32    channels[NUM_CHANNELS] = {0, 1, 2, 3, 4};
ViReal64 ts_secs[MAX_SCANS], ts_usecs[MAX_SCANS];
ViInt32    num_scans;
ViReal64 data[MAX_DATA];
ViInt32    num_data;

// set the scanlist
vtex1048_set_scanlist(vi, channels, NUM_CHANNELS);

// set the trigger source to be timer
vtex1048_set_trig_source_timer(vi, TIMER_INTERVAL);
```

```
// set trigger count
vtex1048_set_trigger_count(vi, MAX_SCANS);

// trigger initialize
vtex1048_init_imm(vi);

// retrieve the data, nominally ready in 2 seconds
vtex1048_read_fifo(vi, MAX_SCANS, ts_secs, ts_usecs, &num_scans, MAX_DATA, data, &num_data,
3);
```

データの相関性

vtex1048_read_fifo 命令で帰ってくるデータは、3 つの 1 次元配列、この場合 data[]、ts_secs[]、ts_usecs[]として出力されます。5 チャンネルのスキャンでは最初のスキャンのデータは data[0]から data[4]までの要素に入り、スキャン開始時間は ts_secs[0] + ts_usecs[0]で表されます。第二スキャンのデータは data[5]から data[9]までの要素に入り、スキャン開始時刻は ts_secs[1] + ts_usecs[1]で表され、第三、第四と同じように続きます。各スキャン内では最初のデータ要素はスキャンリストで最初に宣言したエントリー、第二のデータ要素は二番目に宣言したエントリー、そして第三、第四と続きます。

ひとくちメモ	将来のデータ収集時に最大限のリーディングバッファ容量を確保する為、回収と共に FIFO メモリ内のデータは消去されます。

プログラム例

```
#include <ansi_c.h>
#include <stdio.h>
#include <windows.h>
#include <vtex1048.h>

#define INSTR_RESRC_STR         "TCPIP::192.168.0.127::INSTR"
#define TYPE_T                  3
#define TYPE_E                  4
#define NUM_CHANNELS            11
#define NUM_E_CHANNELS          6
#define NUM_T_CHANNELS          5
#define TRIG_TIMER              0.2     /* (5 readings per sec) */
#define MAX_SCANS               1000
#define MAX_DATA                (NUM_CHANNELS * MAX_SCANS)

int main( int argc, char **argv )
{
   ViSession        vi;
   ViStatus         status;
   /* scanlist entries */
   ViInt32          channels[NUM_CHANNELS] = {0,1,3,6,10,15,24,30,31,32,35};
   ViInt32          e_channels[NUM_E_CHANNELS] = {0,1,3,6,10,15};
   ViInt32          t_channels[NUM_T_CHANNELS] = {24,30,31,32,35};
   ViInt32          i, j;
   ViReal64         ts_secs[MAX_SCANS], ts_usecs[MAX_SCANS];
   ViInt32          numscans;
   ViReal64         data[MAX_DATA];
   ViInt32          numdata;

   /* open a session */
   status = vtex1048_init(INSTR_RESRC_STR, VI_ON, VI_ON, &vi);
   if(status != VI_SUCCESS)
   {
      printf("ERROR OPENING CONNECTION\n");
      return –1;
   }
   printf("Connection opened to %s\n", INSTR_RESRC_STR);

   /* configure the scan list */
   status = vtex1048_set_scanlist(vi, channels, NUM_CHANNELS);
   if(status != VI_SUCCESS)
   {
      printf("ERROR CONFIGURING SCAN LIST\n");
      return –1;
   }

   /* configure the EU conversions */
   status = vtex1048_set_channel_conversion(vi, e_channels, NUM_E_CHANNELS, TYPE_E);
   if(status != VI_SUCCESS)
   {
      printf("ERROR CONFIGURING EU CONVERSIONS\n");
      return –1;
   }
   status = vtex1048_set_channel_conversion(vi, t_channels, NUM_T_CHANNELS, TYPE_T);
   if(status != VI_SUCCESS)
```

```
   {
      printf("ERROR CONFIGURING EU CONVERSIONS\n");
      return –1;
   }

   /* configure the filter frequencies */
   status = vtex1048_set_filt_freq(vi, channels, NUM_CHANNELS, 4.0);
   if(status != VI_SUCCESS)
   {
      printf("ERROR CONFIGURING FILTER FREQUENCIES\n");
      return –1;
   }

   /* configure the FIFO (deg F, blocking mode) */
   status = vtex1048_set_fifo_config(vi, 0, 0, 0, 0, 1);
   if(status != VI_SUCCESS)
   {
      printf("ERROR CONFIGURING FIFO\n");
      return –1;
   }

   /* configure the trigger model */

   /* reset the trigger model to default settings */
   status = vtex1048_reset_trigger_arm(vi);

   /* set the trigger timer */
   status = vtex1048_set_trigger_timer(vi, TRIG_TIMER);
   if(status != VI_SUCCESS)
   {
      printf("ERROR CONFIGURING TIMER\n");
      return –1;
   }

   /* set the trigger count */
   status = vtex1048_set_trigger_count(vi, MAX_SCANS);
   if(status != VI_SUCCESS)
   {
      printf("ERROR CONFIGURING COUNT\n");
      return –1;
   }

   /* initialize the acquisition */
   status = vtex1048_init_imm(vi);
   if(status != VI_SUCCESS)
   {
      printf("ERROR INITIATING TRIGGER\n");
      return –1;
   }

   /* read acquisition data */
   status = vtex1048_read_fifo(vi, MAX_SCANS, ts_secs, ts_usecs, &numscans, MAX_DATA, data, &numdata, (ViInt32)
(MAX_SCANS * TRIG_TIMER + 0.1));
   if(status != VI_SUCCESS)
   {
      printf("ERROR READING DATA\n");
      return –1;
```

```
    }

    /* print acquisition data */
    for(i = 0; i < numscans; i++)
    {
        printf("%.0f.%06.0f: ", ts_secs[i], ts_usecs[i] * 1e6);
        for(j = 0; j < NUM_CHANNELS; j++)
        {
            printf("%6.2f ", data[i * NUM_CHANNELS + j]);
        }
        printf("\n");
    }

    /* close the session */
    status = vtex1048_close(vi);

    return 0;
}
```

第 7 章

命令一覧

はじめに

この章では EX1048 の命令セットを紹介します。最初に、この機器の対応しているすべての命令のアルファベット順のリストとその基本的な説明があります。残りのページでは、それぞれの命令が解説されています。各見出しは、命令またはクエリの文法、パラメータの使用方法と範囲、そして命令の用途が説明されています。

アルファベット順命令リスト

以下は EX1048 で利用可能な命令の概略です。

命令	説明
vtex1048_abort	現在の計測を中断する。
vtex1048_break_lock	測定器のロックを解除する。
vtex1048_check_lock	測定器のロックの状態を尋ねる。
vtex1048_close	測定器のプログラミングセッションを終了する。
vtex1048_get_accum_limit_status	全４８チャンネルの累計のリミットステータスを尋ねる。
vtex1048_get_arm_count	アームの回数を尋ねる。
vtex1048_get_arm_delay	アームの遅延を尋ねる。
vtex1048_get_arm_infinite	無限回アームの有効状態を尋ねる。
vtex1048_get_arm_source	有効なアームソースイベントを尋ねる。
vtex1048_get_channel_conversion	指定したチャンネルの工学単位(EU)変換値を尋ねる。
vtex1048_get_dio_input	デジタル I/O（DIO）ポートの現在の入力ステートを尋ねる。
vtex1048_get_dio_limit_event	有効な DIO のリミットのイベントを尋ねる。
vtex1048_get_dio_limit_event_invert	リミットイベントに指定された DIO チャンネルの反転動作を尋ねる。
vtex1048_get_dio_limit_event_latch	リミットイベントに指定された DIO チャンネルのラッチ動作を尋ねる。
vtex1048_get_dio_output	プログラムされた DIO ポートの出力のステートを尋ねる。
vtex1048_get_dio_output_enable	DIO ポートの出力の有効状態を尋ねる。
vtex1048_get_fifo_config	FIFO メモリーのオーバーフロー動作とデータの書式を尋ねる。
vtex1048_get_fifo_count	FIFO メモリー上のデータページ（スキャン）数を尋ねる。
vtex1048_get_filt_freq	指定したチャンネルのハードウェアフィルタの周波数を尋ねる。
vtex1048_get_init_cont	初期連続モードの有効状態を尋ねる。
vtex1048_get_limit_set0	指定したチャンネルのリミットセット 0 の値を尋ねる。
vtex1048_get_limit_set0_manual	指定したチャンネルのリミットセット 0 の手動入力コントロールを尋ねる。
vtex1048_get_limit_set1	指定したチャンネルのリミットセット 1 の値を尋ねる。
vtex1048_get_scanlist	現在のスキャンリストを尋ねる。
vtex1048_get_trigger_count	トリガの回数を尋ねる。
vtex1048_get_trigger_delay	トリガ遅延を尋ねる。
vtex1048_get_trigger_infinite	無限回トリガーの有効状態を尋ねる。
vtex1048_get_trigger_source	トリガソースイベントの有効状態を尋ねる。
vtex1048_get_trigger_timer	タイマートリガイベントの時間間隔を尋ねる。
vtex1048_get_user_cjc_enable	指定したチャンネルのユーザー定義 CJC 温度の有効状態を尋ねる。

命令	説明
vtex1048_get_user_cjc_temp	指定したチャンネルのユーザー定義 CJC 温度を尋ねる。
vtex1048_get_user_conversion	ユーザー定義の変換多項式を尋ねる。
vtex1048_get_vtb_input	トリガバスの現在の入力状態を尋ねる。
vtex1048_get_vtb_output	トリガバスのプログラムされた出力状態を尋ねる。
vtex1048_get_vtb_output_enable	トリガバスの出力の有効状態を尋ねる。
vtex1048_init	測定器のプログラミングセッションを開始する。
vtex1048_init_imm	トリガを開始する。
vtex1048_lock	測定器のロックを試みる。
vtex1048_read_fifo	測定データを読み出す。
vtex1048_reset	測定器をリセットする。
vtex1048_reset_fifo	FIFO メモリーをクリアする。
vtex1048_reset_trigger_arm	トリガの設定パラメータをリセットする。
vtex1048_revisionQuery	測定器のドライバとファームウェアのバージョンを尋ねる。
vtex1048_self_cal_clear	現在の校正値をクリアする。
vtex1048_self_cal_clear_stored	不揮発性メモリ上の校正値をクリアする。
vtex1048_self_cal_get_status	自己校正の完了状態を尋ねる。
vtex1048_self_cal_init	測定器の自己校正を実行する。
vtex1048_self_cal_is_stored	不揮発性メモリ上の校正値の存在を尋ねる。
vtex1048_self_cal_load	不揮発性メモリ上の校正値を現在の校正値として読み込む。
vtex1048_self_cal_store	現在の校正値を不揮発性メモリに保存する。
vtex1048_set_arm_count	アーム回数を設定する。
vtex1048_set_arm_delay	アーム遅延を設定する。
vtex1048_set_arm_infinite	無限回アームを有効、または無効にする。
vtex1048_set_arm_source	アームソースイベントを設定する。
vtex1048_set_channel_conversion	指定したチャンネルの工学単位(EU) 変換値を設定する。
vtex1048_set_dio_limit_event	リミット評価をデジタル IO ポートの動作と結合する。
vtex1048_set_dio_limit_event_invert	リミットイベントに指定された DIO の反転動作を有効/無効にする。
vtex1048_get_dio_limit_event_latch	リミットイベントに指定された DIO のラッチ動作を有効/無効にする。
vtex1048_get_dio_output	有効な各 DIO ポートの出力レベルを設定する。
vtex1048_get_dio_output_enable	各 DIO チャンネルの出力機能を有効、または無効にする。
vtex1048_set_dio_pulse	DIO ポートの指定したチャンネルから 1 µs のパルスを出力させる。
vtex1048_set_fifo_config	FIFO メモリのデータの書式とオーバーフロー動作を設定する。
vtex1048_set_filt_freq	指定したチャンネルのハードウェアフィルターの周波数を設定する
vtex1048_set_init_cont	初期連続モードを有効、または無効にする。
vtex1048_set_limit_set0	指定したチャンネルのリミットセット 0 の値を手動で設定する。
vtex1048_set_limit_set0_manual	指定したチャンネルのリミットセット 0 の手動入力を有効、無効にする。
vtex1048_set_limit_set1	指定したチャンネルのリミットセット 1 の値を設定する。
vtex1048_set_scanlist	測定のスキャンリストを設定する。
vtex1048_set_trig_source_timer	タイマートリガと時間間隔を設定する。
vtex1048_set_trigger_count	トリガ回数を設定する。
vtex1048_set_trigger_delay	トリガ遅延を設定する。
vtex1048_set_trigger_infinite	無限回トリガを有効、または無効にする。
vtex1048_set_trigger_source	トリガソースイベントを設定する。
vtex1048_set_trigger_timer	タイマーソースイベントの時間間隔を設定する。
vtex1048_set_user_cjc_enable	指定したチャンネルのユーザー定義 CJC 温度を有効または無効にする。
vtex1048_set_user_cjc_temp	指定したチャンネルのユーザー定義 CJC 温度を設定する。
vtex1048_set_user_conversion	ユーザー定義の変換多項式を設定する。
vtex1048_set_vtb_output	有効なトリガバスの各チャンネルの出力レベルを設定する。
vtex1048_set_vtb_output_enable	トリガバスの各チャンネルの出力機能を有効または無効にする。
vtex1048_set_vtb_pulse	トリガバスの指定したチャンネルに 1 µs のパルスを発生する。
vtex1048_soft_arm	ソフトウェアアームを実行する。
vtex1048_soft_trigger	ソフトウェアトリガを実行する。

命令	説明
vtex1048_unlock	測定器のロックを解除する。

命令一覧

この章の残りは各命令の解説です。命令の用法を定義するため、以下の事項が説明されています。

用途	命令の用途を説明。
種類	イベントや設定などの、命令の種類の説明。
命令文法	命令の書式の詳しい説明。
パラメータ	命令のパラメータの意味と有効範囲の説明。
リセット値	**vtex1048_reset** 命令後の設定値の説明。
クエリに対する応答	クエリ応答の書式と応答値の範囲の説明。
解説	命令の動作の詳しい説明とその他参考資料
例	各命令とクエリの正しい使用例。

vtex1048_abort

用途	現在の計測を中断する。
種類	イベント
命令文法	vtex1048_abort(**vi**)
パラメータ	**vi** = セッション ID
リセット値	N/A
クエリに対する応答	N/A
解説	現在の計測を中断し、トリガモデルを IDLE のレイヤーに戻す。
例	

vtex1048_break_lock

用途	測定器のロックを解除する
種類	イベント
命令文法	vtex1048_break_lock(**vi**)
パラメータ	**vi** = セッション ID
リセット値	N/A
クエリに対する応答	N/A
解説	測定器のロックを解除する。　ロックの所有者に関係なく、ロックが解除されます。これにより、ロックした IP アドレスが無効になった場合、測定器を回復することができます。 **注：測定器のロックを解除しても、自動的にロックが与えられるわけではありません。別に *vtex1048_lock* の命令を実行する必要があります。**
例	

vtex1048_check_lock

用途	測定器のロックの状態を尋ねる。
種類	クエリ
命令文法	vtex1048_check_lock(**vi, &locked, &mine**)
パラメータ	**vi** = セッション ID **locked** = 測定器のロック状態を示す論理値。 **mine** = クエリを実行したホスト IP アドレスがロックを所有しているかどうかを示す論理値。
リセット値	N/A
クエリに対する応答	N/A
解説	測定器のロックの状態を尋ねる。ロックされた状態のときに、EX1048 はロックの命令を送ったホスト IP アドレスからの命令だけを受け付けます。
例	

vtex1048_close

用途	測定器のプログラミングセッションを終了する。
種類	イベント
命令文法	vtex1048_close(**vi**)
パラメータ	**vi** = セッション ID
リセット値	N/A
クエリに対する応答	N/A
解説	測定器のプログラミングセッションを終了する。　この命令は、アプリケーションの終了時に実行される必要があります。　動作の一部として、測定器のロックを解除し、次のアプリケーションのために測定器を適切な状態に戻します。
例	

vtex1048_get_accum_limit_status

用途	全４８チャンネルの累計のリミットステータスを尋ねる。
種類	クエリ
命令文法	vtex1048_get_accum_limit_status(**vi**, **set0_lower[]**, **set0_upper[]**, **set1_lower[]**, **set1_upper[]**)
パラメータ	**vi** = セッション ID **set0_lower[]** = 各チャンネルのリミットセット 0 の下限に対応する、全 48 チャンネルの累積リミット状態を表す論理値。 チャンネル 0 から 47 は配列要素の[0]から[47]に対応する。 **set0_upper[]** = 各チャンネルのリミットセット 0 の上限に対応する、全 48 チャンネルの累積リミット状態を表す論理値。 チャンネル 0 から 47 は配列要素の[0]から[47]に対応する。 **set1_lower[]** = 各チャンネルのリミットセット 1 の下限に対応する、全 48 チャンネルの累積リミット状態を表す論理値。 チャンネル 0 から 47 は配列要素の[0]から[47]に対応する **set1_upper[]** = 各チャンネルのリミットセット 1 の上限に対応する、全 48 チャンネルの累積リミット状態を表す論理値。 チャンネル 0 から 47 は配列要素の[0]から[47]に対応する
リセット値	N/A
クエリに対する応答	**set0_lower**: 各配列要素は 0 (限界内)か 1 (限界超過) **set0_upper**: 各配列要素は 0 (限界内)か 1 (限界超過) **set1_lower**: 各配列要素は 0 (限界内)か 1 (限界超過) **set1_upper**: 各配列要素は 0 (限界内)か 1 (限界超過)
解説	全４８チャンネルの累計のリミットステータスを尋ねる。 スキャンリストに含まれているか否かにかかわらず、すべてのチャンネルのリミットの状態が返されます。 返された値は、各チャンネルごとに、最後のトリガーの初期化以来、測定データが、リミット値を一度でも超過したかどうかを表します。 *vtex1048_init_imm* 命令の一部で、リミット状態はクリアされます。 スキャンされないチャンネルのリミットステータスは常に 0 になります。
例	

vtex1048_get_arm_count

用途	アームの回数を尋ねる。
種類	クエリ
命令文法	vtex1048_get_arm_count(**vi**, **&count**)
パラメータ	**vi** = セッション ID **count** = アーム回数を示す整数値出力
リセット値	N/A
クエリに対する応答	**count**: 1 から(2^{31}-1)までの整数値
解説	アームの回数を尋ねる。
例	

vtex1048_get_arm_delay

用途	アームの遅延を尋ねる。
種類	クエリ
命令文法	vtex1048_get_arm_delay(**vi**, **&arm_delay**)
パラメータ	**vi** = セッション ID **arm_delay** = アーム遅延を表す秒単位の実数値。
リセット値	N/A
クエリに対する応答	**arm_delay**: 0 から 4294 までの実数
解説	アームの遅延を尋ねる。アームイベントを検出してから、トリガモデルのトリガレイヤーに移行するまでの時間。
例	

vtex1048_get_arm_infinite

用途	無限回アームの有効状態を尋ねる。
種類	クエリ
命令文法	vtex1048_get_arm_infinite(**vi**, **&arminf**)
パラメータ	**vi** = セッション ID **arminf** = 無限回アームの有効状態を示す論理値
リセット値	N/A
クエリに対する応答	**arminf**: 0 または 1
解説	無限回アームの有効状態を尋ねる。
例	

vtex1048_get_arm_source

用途	有効なアームソースイベントを尋ねる。
種類	クエリ
命令文法	vtex1048_get_arm_source(**vi**,**vtb_masks[]**,**dio_masks[]**,**&timer_enable**,**&immediate**)
パラメータ	**vi** = セッション ID **vtb_masks[]** = ８チャンネルのトリガバスのアームイベントの有効状態を示す８ビットの整数値４個分の配列。値の順番は、上がりエッジ、下りエッジ、正のレベル、負のレベル。　８ビットのうち MSB は VTB チャンネル 7,　LSB は VTB チャンネル 0 に対応。 **dio_masks[]** = ８チャンネルのデジタル IO ポートのアームイベントの有効状態を示す８ビットの整数値４個分の配列。値の順番は、上がりエッジ、下りエッジ、正のレベル、負のレベル。　８ビットのうち MSB は DIO チャンネル 7,　LSB は DIO チャンネル 0 に対応。 **timer_enable** = アームイベントとしてタイマーが有効かどうかを示す論理値。 **immediate** = アームイベントとして即時アームが有効かどうかを示す論理値。
リセット値	N/A
クエリに対する応答	**vtb_masks**: 0 から 255　までの整数の配列 **dio_masks**: 0 から 255　までの整数の配列 **timer_enable**: 0 or 1 **immediate**: 0 or 1
解説	有効なアームソースイベントを尋ねる。応答の値にかかわらず、ソフトウェアによるアームは常に有効です。
例	

vtex1048_get_channel_conversion

用途	指定したチャンネルの工学単位(EU)変換値を尋ねる。
種類	クエリ
命令文法	vtex1048_get_channel_conversion(**vi**, **channel**, **&eu_conv**)
パラメータ	**vi** = セッション ID **channel** = 工学単位変換値を尋ねるチャンネル。 0 から 47 までの整数値。 **eu_conv** = 工学単位変換を表す整数値出力。 値は以下の関数に対応。 **整数値** / **関数** 0 Voltage 1 Type J 2 Type K 3 Type T 4 Type E 5 Type B 6 Type S 7 Type R 8 Type N 9 User-defined 0 10 User-defined 1
リセット値	N/A
クエリに対する応答	**eu_conv**: 0 から 10 までの整数値。
解説	指定したチャンネルの工学単位(EU)変換値を尋ねる。
例	

vtex1048_get_dio_input

用途	デジタル I/O (DIO)ポートの現在の入力ステートを尋ねる。
種類	クエリ
命令文法	vtex1048_get_dio_input(**vi**, **&dio_in**)
パラメータ	**vi** = セッション ID **dio_in** = ポートの８ビットの値をあらわす整数。　８ビットのうち MSB は DIO のチャンネル７、LSB は DIO のチャンネル 0 に対応します。
リセット値	N/A
クエリに対する応答	**dio_in**: 0 から 255 までの整数値。
解説	デジタル I/O (DIO)ポートの現在の入力ステートを尋ねる。

例	`// check state of DIO bits 7, 4, and 0`
	`ViInt32 dio_in;`
	`vtex1048_get_dio_input(vi, &dio_in);`
	`if (dio_in & 0x80)`
	`printf( “Bit 7 is high” );`
	`else printf( “Bit 7 is low” );`
	`if (dio_in & 0x10)`
	`printf( “Bit 4 is high” );`
	`else printf( “Bit 4 is low” );`
	`if (dio_in & 0x01)`
	`printf( “Bit 0 is high” );`
	`else printf( “Bit 0 is low” );`

vtex1048_get_dio_limit_event

用途	有効な DIO のリミットのイベントを尋ねる。
種類	クエリ
命令文法	vtex1048_get_dio_limit_event(**vi**, **dio_channel**, **limit_masks[]**)
パラメータ	**vi** = セッション ID **dio_channel** = DIO のリミットイベントの状態を尋ねるチャンネル。 0 から 7 までの整数。 **limit_masks[]** = 各チャンネルごとに、４種類のリミット条件をあらわす、４８個の４ビットの整数配列。 ４ビットのうち、順番は、リミットセット 0 の下限、リミットセット 0 の上限、リミットセット 1 の下限、リミットセット 1 の上限。 チャンネル 0 から 47 は配列の要素[0]から[47]に対応。
リセット値	N/A
クエリに対する応答	**limit_masks**: 配列の各要素は 0 から 15 までの整数値。
解説	有効な DIO のリミットのイベントを尋ねる。
例	

vtex1048_get_dio_limit_event_invert

用途	リミットイベントに指定された DIO チャンネルの反転動作を尋ねる。
種類	クエリ
命令文法	vtex1048_get_dio_limit_event_invert(**vi, dio_channel, &invert**)
パラメータ	**vi** = セッション ID **dio_channel** = デジタル I/O ポートチャンネル。 0 から 7 までの整数。 **invert** = 指定した DIO のチャンネルが反転モードで動作しているかどうかを表す論理値。
リセット値	N/A
クエリに対する応答	**invert**: 0 または 1
解説	リミットイベントに指定された DIO チャンネルの反転動作を尋ねる。
例	

vtex1048_get_dio_limit_event_latch

用途	リミットイベントに指定された DIO チャンネルのラッチ動作を尋ねる。
種類	クエリ
命令文法	vtex1048_get_dio_limit_event_latch(**vi, dio_channel, &latch**)
パラメータ	**vi** = セッション ID **dio_channel** = DIO ポートのチャンネル。 0 から 7 までの整数。 **latch** = 指定した DIO チャンネルがラッチモードかどうかを示す論理値。
リセット値	N/A
クエリに対する応答	**latch**: 0 または 1
解説	リミットイベントに指定された DIO チャンネルのラッチ動作を尋ねる。
例	

vtex1048_get_dio_output

用途	プログラムされた DIO ポートの出力のステートを尋ねる。
種類	クエリ
命令文法	vtex1048_get_dio_output(**vi**, **&dio_out**)
パラメータ	**vi** = セッション ID **dio_out** = ８ビットポートのプログラム出力のステートをあらわす整数値。　８ビットのフィールドのうち MSB は DIO のチャンネル７、LSB は DIO のチャンネル 0 に対応します。
リセット値	N/A
クエリに対する応答	**dio_out**: 0 から 255 までの整数値
解説	プログラムされた DIO ポートの出力のステートを尋ねる。この命令はプログラムの設定を返すだけです。　実際には出力を有効にする必要があるので、必ずしも実際の出力のステートを表しているわけではありません。

例	// query status of DIO bit 4
	ViInt32 dio_out;
	ViInt32 dio_outen;
	vtex1048_get_dio_output(vi, &dio_out);
	vtex1048_get_dio_output_enable(vi, &dio_outen);
	if (dio_out & 0x10)
	printf(“Bit 4 is set high”);
	else printf(“Bit 4 is set low”);
	if (dio_outen & 0x10)
	printf(“Bit 4 is enabled”);
	else printf(“Bit 4 is not enabled”);

vtex1048_get_dio_output_enable

用途	DIO ポートの出力の有効状態を尋ねる。
種類	クエリ
命令文法	vtex1048_get_dio_output_enable(**vi**, **&out_enable**)
パラメータ	**vi** = セッション ID **out_enable** = ８ビットポートの出力の有効状態をあらわす整数値。　８ビットのフィールドのうち MSB は DIO のチャンネル７、LSB は DIO のチャンネル０に対応します。
リセット値	N/A
クエリに対する応答	**out_enable**: 0 から 255 までの整数値。
解説	DIO ポートの出力の有効状態を尋ねる。

例	`// query status of DIO bit 4`
	`ViInt32 dio_out;`
	`ViInt32 dio_outen;`
	`vtex1048_get_dio_output(vi, &dio_out);`
	`vtex1048_get_dio_output_enable(vi, &dio_outen);`
	`if (dio_out & 0x10)`
	`printf( “Bit 4 is set high” );`
	`else printf( “Bit 4 is set low” );`
	`if (dio_outen & 0x10)`
	`printf( “Bit 4 is enabled” );`
	`else printf( “Bit 4 is not enabled” );`

vtex1048_get_fifo_config

用途	FIFO メモリーのオーバーフロー動作とデータの書式を尋ねる。
種類	クエリ
命令文法	vtex1048_get_fifo_config(**vi, &report_cjc, &report_timestamp, &report_celsius, &blocking_mode**)
パラメータ	**vi** = セッション ID **report_cjc** = 測定データに CJC 温度がレポートされるかどうかを示す論理値。 CJC 温度は、**report_celsius** パラメータの値にかかわらず常に °C の単位であらわされます。 **report_timestamp** = 測定データにチャンネルごとのタイムスタンプがレポートされるかどうかを示す論理値 **report_celsius** = 入力チャンネルのデータの単位が °C(真) か °F(偽)をあらわす論理値。 **blocking_mode** = スキャニング中に最大容量に達したときの FIFO の動作を示す論理値。ブロッキングモードが有効な場合は、それ以降の測定値は排除され、バッファの内容が維持されます。 ブロッキングモードが無効な場合は、サーキュラーバッファとして働き、古いデータは新しいデータに上書きされます。
リセット値	N/A
クエリに対する応答	**report_cjc**: 0 または 1 **report_timestamp**: 0 または 1 **report_celsius**: 0 または 1 **blocking_mode**: 0 または 1
解説	FIFO メモリのオーバーフロー動作とデータの書式を尋ねる。 **注: report_cjc と report_timestamp パラメータの値にかかわらず, CJC 温度とタイムスタンプの変化分の情報は *vtex1048_read_fifo* の命令ではアクセスできません。これらの情報は、ストリーミングインターフェースと相当するパラメータによって得ることができます。**
例	

vtex1048_get_fifo_count

用途	FIFO メモリ上のデータページ（スキャン）数を尋ねる。
種類	クエリ
命令文法	vtex1048_get_fifo_count(**vi**, **&count**)
パラメータ	**vi** = セッション ID **count** = データページ数を表す整数出力
リセット値	N/A
クエリに対する応答	**count**: 0 から 187246 までの整数値。
解説	FIFO メモリ上のデータページ（スキャン）数を尋ねる。
例	

vtex1048_get_filt_freq

用途	指定したチャンネルのハードウェアフィルターの周波数を尋ねる。
種類	クエリ
命令文法	vtex1048_get_filt_freq(**vi**, **channel**, **&filt_freq**)
パラメータ	**vi** = セッション ID **channel** = ハードウェアフィルターの周波数を尋ねるチャンネル。 0 から 47 までの整数値。 **filt_freq** = 周波数をあらわす実数値。
リセット値	N/A
クエリに対する応答	**filt_freq**: 4.0 から 1000.0 までの実数値。
解説	指定したチャンネルのハードウェアフィルターの周波数を尋ねる。
例	

vtex1048_get_init_cont

用途	初期連続モードの有効状態を尋ねる。
種類	クエリ
命令文法	vtex1048_get_init_cont(**vi**, **&init_cont_mode**)
パラメータ	**vi** = セッション ID **init_cont_mode** = 初期連続モードが有効かどうかを示す論理値。
リセット値	N/A
クエリに対する応答	**init_cont_mode**: 0 または 1
解説	初期連続モードの有効状態を尋ねる。
例	

vtex1048_get_limit_set0

用途	指定したチャンネルのリミットセット 0 の値を尋ねる。
種類	クエリ
命令文法	vtex1048_get_limit_set0(**vi, channel, &lower_limit, &upper_limit**)
パラメータ	**vi** = セッション ID **channel** = リミットセット 0 の値を尋ねるチャンネル。　0 から 47 までの整数値。 **lower_limit** = 下限を示す実数値。 **upper_limit** = 上限を示す実数値。
リセット値	N/A
クエリに対する応答	**lower_limit**: -3e+38 から 3e+38. までの実数値。 **upper_limit**: -3e+38 から 3e+38. までの実数値。
解説	指定したチャンネルのリミットセット 0 の値を尋ねる。
例	

vtex1048_get_limit_set0_manual

用途	指定したチャンネルのリミットセット 0 の手動入力コントロールを尋ねる。
種類	クエリ
命令文法	vtex1048_get_limit_set0_manual(**vi**, **channel**, **&manual**)
パラメータ	**vi** = セッション ID **channel** = 手動入力コントロールの値を尋ねるチャンネル。 0 から 47 までの整数値。 **manual** = 手動入力が有効かどうかを表す論理値。
リセット値	N/A
クエリに対する応答	**manual**: 0 か 1
解説	指定したチャンネルのリミットセット 0 の手動入力コントロールを尋ねる。
例	

vtex1048_get_limit_set1

用途	指定したチャンネルのリミットセット 1 の値を尋ねる。
種類	クエリ
命令文法	vtex1048_get_limit_set1(**vi, channel, &lower_limit, &upper_limit**)
パラメータ	**vi** = セッション ID **channel** = リミットセット 1 の値を尋ねるチャンネル。 0 から 47 までの整数値。 **lower_limit** = 下限を表す実数出力。 **upper_limit** = 上限を表す実数出力。
リセット値	N/A
クエリに対する応答	**lower_limit**: -3e+38 から 3e+38 までの実数値。 **upper_limit**: -3e+38 から 3e+38 までの実数値。
解説	指定したチャンネルのリミットセット 1 の値を尋ねる。
例	

vtex1048_get_scanlist

用途	現在のスキャンリストを尋ねる。
種類	クエリ
命令文法	vtex1048_get_scanlist(**vi**, **channels[]**, **&numChannels**)
パラメータ	**vi** = セッション ID **channels[]** = スキャンリストを受け取る配列。 **numChannels** = スキャンリストのチャンネル数を示す整数出力。
リセット値	N/A
クエリに対する応答	**channels[]**: 0 から 47 までの整数値の入る、最高 48 要素の配列。 **numChannels**: 1 から 48 までの整数値。
解説	現在のスキャンリストを尋ねる。
例	

vtex1048_get_trigger_count

用途	トリガの回数を尋ねる。
種類	クエリ
命令文法	vtex1048_get_trigger_count(**vi**, **&count**)
パラメータ	**vi** = セッション ID **count** = トリガ回数を表す整数値
リセット値	N/A
クエリに対する応答	**count**: 1 から (2^31-1)までの整数値。
解説	トリガの回数を尋ねる。
例	

vtex1048_get_trigger_delay

用途	トリガ遅延を尋ねる。
種類	クエリ
命令文法	vtex1048_get_trigger_delay(**vi**, **&trig_delay**)
パラメータ	**vi** = セッション ID **trig_delay** = トリガ遅延を示す秒単位の実数
リセット値	N/A
クエリに対する応答	**trig_delay**: 0 から 4294　までの実数値。
解説	トリガ遅延を尋ねる。トリガイベントの検出からスキャンリストを実行するまでの遅延時間。
例	

vtex1048_get_trigger_infinite

用途	無限回トリガの有効状態を尋ねる。
種類	クエリ
命令文法	vtex1048_get_trigger_infinite(**vi**, **&triginf**)
パラメータ	**vi** = セッション ID **triginf** = 無限回トリガが有効かどうかを示す論理値。
リセット値	N/A
クエリに対する応答	**triginf**: 0 または 1
解説	無限回トリガの有効状態を尋ねる。
例	

vtex1048_get_trigger_source

用途	トリガソースイベントの有効状態を尋ねる。
種類	クエリ
命令文法	vtex1048_get_trigger_source(**vi**, **vtb_masks[]**, **dio_masks[]**, **&timer_enable**, **&immediate**)
パラメータ	**vi** = セッション ID **vtb_masks[]** = ８チャンネルトリガバスからのトリガーイベントの有効状態を示す８ビット整数４個分の配列。数値の順番は：上りエッジ、下りエッジ、正のレベル、負のレベル。　８ビットのフィールドのうち MSB は VTB チャンネル７、LSB は VTB チャンネル 0 に対応。 **dio_masks[]** = ８チャンネルの DIO ポートのトリガイベントの有効状態を示す８ビットの整数値４個分の配列。　数値の順番は：上りエッジ、下りエッジ、正のレベル、負のレベル。　８ビットのフィールドのうち MSB は DIO チャンネル７、LSB は DIO チャンネル 0 に対応。 **timer_enable** = タイマートリガが有効かどうかを示す論理値。 **immediate** = トリガイベントとして即時トリガが有効かどうかを示す論理値。
リセット値	N/A
クエリに対する応答	**vtb_masks**: 各配列の要素は 0 から 255 までの整数。 **dio_masks**:　各配列の要素は 0 から 255 までの整数。 **timer_enable**: 0 または 1 **immediate**: 0 または 1
解説	トリガソースイベントの有効状態を尋ねる。　応答値にかかわらず、ソフトウェアのトリガは常に有効です。
例	

vtex1048_get_trigger_timer

用途	タイマートリガイベントの時間間隔を尋ねる。
種類	クエリ
命令文法	vtex1048_get_trigger_timer(**vi**, **&trig_timer**)
パラメータ	**vi** = セッション ID **trig_timer** =タイマーイベントの秒単位の時間間隔をあらわす実数値
リセット値	N/A
クエリに対する応答	**trig_timer**: 0.001 から 4294 までの実数値
解説	タイマートリガイベントの時間間隔を尋ねる。　アームとトリガイベントの両方に同じ値が使われます。
例	

vtex1048_get_user_cjc_enable

用途	指定したチャンネルのユーザー定義 CJC 温度の有効状態を尋ねる。
種類	クエリ
命令文法	vtex1048_get_user_cjc_enable(**vi**, **channel**, **&usercjc**)
パラメータ	**vi** = セッション ID **channel** = 有効の状態を尋ねるチャンネル。 0 から 47 までの整数値。 **usercjc** = ユーザー定義の CJC 温度が有効かどうかを示す論理値。
リセット値	N/A
クエリに対する応答	**usercjc**: 0 または 1
解説	指定したチャンネルのユーザー定義 CJC 温度の有効状態を尋ねる。
例	

vtex1048_get_user_cjc_temp

用途	指定したチャンネルのユーザー定義 CJC 温度を尋ねる。
種類	クエリ
命令文法	vtex1048_get_user_cjc_temp(**vi**, **channel**, **&cjc_temp**)
パラメータ	**vi** = セッション ID **channel** = ユーザー定義の CJC 温度を尋ねるチャンネル。0 から 47 までの整数値 **cjc_temp** = °C 単位で CJC 温度 をあらわす実数値。
リセット値	N/A
クエリに対する応答	**cjc_temp**: -3e+38 から 3e+38. のあいだの実数値
解説	指定したチャンネルのユーザー定義 CJC 温度を尋ねる。
例	

vtex1048_get_user_conversion

用途	ユーザー定義の変換多項式を尋ねる。
種類	クエリ
命令文法	vtex1048_get_user_conversion(**vi, eu_conv, fwdcoeff[], invcoeff[]**)
パラメータ	**vi** = セッション ID **eu_conv** = 多項式セット　9（ユーザー0）または 10（ユーザー1） **fwdcoeff[]** = 正変換多項式の係数。　係数 c_0 から c_{12} までは配列要素［0］から［12］に対応。 **invcoeff[]** = 逆変換多項式の係数。　係数 d_0 から d_{12} までは配列要素［0］から［12］に対応。
リセット値	N/A
クエリに対する応答	**fwdcoeff[]**: 各配列要素は -3e+38 と 3e+38 の間の実数値。 **invcoeff[]**: 各配列要素は -3e+38 と 3e+38 の間の実数値。
解説	ユーザー定義の変換多項式を尋ねる。　*vtex1048_set_user_conversion* の命令で指定した数にかかわらず、常に 12 要素すべての値が配列に返されます。
例	

vtex1048_get_vtb_input

用途	トリガバスの現在の入力状態を尋ねる。
種類	クエリ
命令文法	vtex1048_get_vtb_input(**vi**, **&vtb_in**)
パラメータ	**vi** = セッション ID **vtb_in** = ポートの８ビットの値をあらわす整数値。　８ビットのうち MSB は VTB のチャンネル 7, LSB　は VTB のチャンネル 0 に対応。
リセット値	N/A
クエリに対する応答	**vtb_in**: 0 から 255 までの整数値。
解説	トリガバスの現在の入力状態を尋ねる。

例	// check state of VTB bits 7, 4, and 0
	ViInt32 vtb_in;
	vtex1048_get_vtb_input(vi, &vtb_in);
	if (vtb_in & 0x80)
	printf(“Bit 7 is high”);
	else printf(“Bit 7 is low”);
	if (vtb_in & 0x10)
	printf(“Bit 4 is high”);
	else printf(“Bit 4 is low”);
	if (vtb_in & 0x01)
	printf(“Bit 0 is high”);
	else printf(“Bit 0 is low”);

vtex1048_get_vtb_output

用途	トリガバスのプログラムされた出力状態を尋ねる。
種類	クエリ
命令文法	vtex1048_get_vtb_output(**vi**, **&vtb_out**)
パラメータ	**vi** = セッション ID **vtb_out** = 8 ビットポートのプログラム出力状態を表す整数値。８ビットのうち MSB は VTB チャンネル 7, LSB は VTB チャンネル 0 に対応。
リセット値	N/A
クエリに対する応答	**vtb_out**: 0 から 255 までの整数値
解説	トリガバスのプログラムされた出力状態を尋ねる。この命令はプログラムの設定を返すだけです。 実際には出力を有効にする必要があるので、必ずしも実際の出力のステートを表しているわけではありません。

例	// query status of VTB bit 4
	ViInt32 vtb_out;
	ViInt32 vtb_outen;
	vtex1048_get_vtb_output(vi, &vtb_out);
	vtex1048_get_vtb_output_enable(vi, &vtb_outen);
	if (vtb_out & 0x10)
	printf(“Bit 4 is set high”);
	else printf(“Bit 4 is set low”);
	if (vtb_outen & 0x10)
	printf(“Bit 4 is enabled”);
	else printf(“Bit 4 is not enabled”);

vtex1048_get_vtb_output_enable

用途	トリガバスの出力の有効状態を尋ねる。
種類	クエリ
命令文法	vtex1048_get_vtb_output_enable(**vi**, **&out_enable**)
パラメータ	**vi** = セッション ID **out_enable** = ８ビットポートの出力のステートをあらわす整数値。　８ビットのフィールドのうち MSB は VTB のチャンネル７、LSB は VTB のチャンネル 0 に対応します。
リセット値	N/A
クエリに対する応答	**out_enable**: 0 から 255 までの整数値
解説	トリガバスの出力の有効状態を尋ねる。

例	// query status of VTB bit 4
	ViInt32 vtb_out;
	ViInt32 vtb_outen;
	vtex1048_get_vtb_output(vi, &vtb_out);
	vtex1048_get_vtb_output_enable(vi, &vtb_outen);
	if (vtb_out & 0x10)
	printf(“Bit 4 is set high”);
	else printf(“Bit 4 is set low”);
	if (vtb_outen & 0x10)
	printf(“Bit 4 is enabled”);
	else printf(“Bit 4 is not enabled”);

vtex1048_init

用途	測定器のプログラミングセッションを開始する。
種類	イベント
命令文法	vtex1048_init(**resourceName, IDQuery, resetDevice, &vi**)
パラメータ	**resourceName** = VISA リソース文字列。 書式は、 “TCPIP::<W.X.Y.Z>::INSTR”, ここで、 W.X.Y.Z は接続する IP アドレスをあらわす。 **IDQuery** = 接続した測定器が EX1048 であることの確認動作を実行するかどうかを表す論理値。 **resetDevice** = 接続時に測定器をリセットするかどうかを示す論理値。 **vi** = 各接続ごとに固有のセッションハンドル(ID)。
リセット値	N/A
クエリに対する応答	N/A
解説	測定器のプログラミングセッションを開始する。 EX1048 と交信するためには、この命令がエラー無く実行される必要があります。 同一のアプリケーションから、複数の測定器へのセッションを同時に開始することができます。 それぞれのセッションは、独自のセッションハンドルで識別されます。

例	#include <vtex1048.h>
	
	#define INSTR_RESRC_STR “TCPIP::192.168.0.127::INSTR”
	
	ViSession vi;
	
	// open a session to EX1048 at IP address 192.168.0.127
	vtex1048_init(INSTR_RESRC_STR, 1, 1, &vi);

vtex1048_init_imm

用途	トリガを開始する。
種類	イベント
命令文法	vtex1048_init_imm(**vi**)
パラメータ	**vi** = セッション ID
リセット値	N/A
クエリに対する応答	N/A
解説	トリガを開始する。トリガモデルを IDLE レイヤーから移行する。
例	

vtex1048_lock

用途	測定器のロックを試みる。
種類	イベント
命令文法	vtex1048_lock(**vi**)
パラメータ	**vi** = セッション ID
リセット値	N/A
クエリに対する応答	N/A
解説	測定器のロックを試みる。　ロックされた EX1048 は、ロックコマンドを実行した、ホスト IP アドレスからの命令だけを受け付けます。　測定器が、まだ、ほかのユーザーからロックされていない場合に限り、ロックすることができます。 設計上、ロッキングの動作は、*vtex1048_break_lock* の命令を実行した、別のホストから解除することができます。　そのため、ロックはほかのユーザに対して、測定器が保護状態で動作しているが、完全に保証されているわけではないという警告をします。 測定器のろっくの状態は、*vtex1048_reset* の命令には影響されません。 自己校正をするには、事前にロックする必要があります。
例	

vtex1048_read_fifo

用途	測定データを測定器から読み出す。
種類	クエリ
命令文法	vtex1048_read_fifo(**vi,maxscans,ts_secs[],ts_usecs[],&numscans,maxdata,data[], &numdata,to_secs**)
パラメータ	**vi** = セッション ID **maxscans** = 読み出すスキャンの最大数。 1 から (2^31-1)までの整数。 **ts_secs[]** = スキャンの開始時刻、(Jan. 1, 1970)からの秒数、を読み込む配列。 **ts_usecs[]** = スキャンの開始時刻、残りのマイクロ秒数。 **numscans** = 実際に読み出されたスキャン数をあらわす整数。 **maxdata** = 読み出すデータ配列の最大長。 1 から (2^31-1)までの整数。 **data[]** = 読み出すサンプルデータの配列。 **numdata** = 実際に読み出されたデータのサンプル数を表す整数。 **to_secs** = EX1048 からデータを読み込む際のタイムアウト値（秒単位）。 0 から (2^31-1)の整数で、0 は無限時間を意味する。
リセット値	N/A
クエリに対する応答	**ts_secs[]**: 実数の配列。 **ts_usecs[]**: 実数の配列。 **numscans**: 0 から (2^31-1)までの配列。 **data[]**: 実数の配列。 **numdata**: 0 から (2^31-1)までの整数。
解説	測定データを読み出す。 入力チャンネルの測定データとスキャン開始時間が読み出される。 **注: 測定データの読み出しバッファ容量を最大に保つため、読み出されたデータは、FIFO メモリから消去されます。**

例	ViInt32 channels[5] = {0, 1, 2, 3, 4};
	ViReal64 ts_secs[20], ts_usecs[20], data[100];
	ViInt32 num_scans, num_data;
	
	vtex1048_set_scanlist(vi, channels, 5);
	vtex1048_set_trig_source_timer(vi, 0.01);
	vtex1048_set_trigger_count(vi, 20);
	vtex1048_init_imm(vi);
	vtex1048_read_fifo(vi, 20, ts_secs, ts_usecs, &num_scans, 100, data, &num_data, 3);

vtex1048_reset

用途	測定器をリセットする。
種類	イベント
命令文法	vtex1048_reset(**vi**)
パラメータ	**vi** = セッション ID
リセット値	N/A
クエリに対する応答	N/A
解説	測定器をリセットする。EX1048 のすべての測定設定パラメータを初期設定値に戻す。 **注: 測定器をリセットすると FIFO メモリ内のデータは消去されます。 リセットを実行する前に、必要な FIFO 内の測定データを読み出しておかなければなりません。**
例	

vtex1048_reset_fifo

用途	FIFO メモリをクリアする。
種類	イベント
命令文法	vtex1048_reset_fifo(**vi**)
パラメータ	**vi** = セッション ID
リセット値	N/A
クエリに対する応答	N/A
解説	FIFO メモリをクリアする。トリガ開始命令を受けると、メモリは消去され、データの読み込むことで、メモリが使えるようになるので、この命令は通常必要ありません。つまり、新しく測定をする際、あえて FIFO をクリアする必要はありません。
例	

vtex1048_reset_trigger_arm

用途	トリガーの設定パラメータをリセットする。
種類	イベント
命令文法	vtex1048_reset_trigger_arm(**vi**)
パラメータ	**vi** = セッション ID
リセット値	N/A
クエリに対する応答	N/A
解説	他の測定パラメータに影響を与えずに、トリガの設定パラメータだけを初期設定値にリセットする。　一方、 *vtex1048_reset* 命令はすべての測定パラメータを初期設定値にリセットします。
例	

vtex1048_revisionQuery

用途	測定器のドライバとファームウェアのバージョンを尋ねる。
種類	クエリ
命令文法	vtex1048_revisionQuery(**vi,driverRev[],instrRev[]**)
パラメータ	**vi** = セッション ID **driverRev[]** = 測定器ドライバーのリリースバージョンを返す配列。 **instrRev[]** = 内臓ファームウェアのリリースバージョンを返す配列。
リセット値	N/A
クエリに対する応答	**driverRev[]**: 文字列。 **instrRev[]**: 文字列。
解説	測定器のドライバとファームウェアのバージョンを尋ねる。
例	ViChar driverRev[256], instrRev[256]; vtex1048_revisionQuery(vi, driverRev, instrRev);

vtex1048_self_cal_clear

用途	現在の校正値をクリアする。
種類	イベント
命令文法	vtex1048_self_cal_clear(**vi**)
パラメータ	**vi** = セッション ID
リセット値	N/A
クエリに対する応答	N/A
解説	現在の校正値をクリアする。　この動作は、揮発性メモリ上の校正値をクリアしますが、不揮発性メモリに保存した校正値データには影響しません。
例	

vtex1048_self_cal_clear_stored

用途	不揮発性メモリ上の校正値をクリアする。
種類	イベント
命令文法	vtex1048_self_cal_clear_stored(**vi**)
パラメータ	**vi** = セッション ID
リセット値	N/A
クエリに対する応答	N/A
解説	不揮発性メモリ上の校正値をクリアする。　この命令では、不揮発性メモリ上の校正値だけがクリアされ、現在の校正値は消去されません。　不揮発性メモリ上に校正値が無いときにこの命令が実行されるとエラーが発生します。
例	

vtex1048_self_cal_get_status

用途	自己校正の完了状態を尋ねる。
種類	クエリ
命令文法	vtex1048_self_cal_get_status(**vi, &cal_percent**)
パラメータ	**vi** = セッション ID **cal_percent** = 自己校正が何パーセント完了したかを示す十進法の整数値。
リセット値	N/A
クエリに対する応答	**cal_percent**: 0 から 100 までの整数。
解説	自己校正の完了状態を尋ねる。 **注: 100 パーセント完了するまでは、測定器に次のコマンドを送らないでください。**
例	

vtex1048_self_cal_init

用途	測定器の自己校正を実行する。
種類	イベント
命令文法	vtex1048_self_cal_init(**vi, &override**)
パラメータ	**vi** = セッション ID **override** = オーバーライドコードをあらわす十進数。 EX1048 が 60 分以上連続して、電源が入った状態になる前に自己校正を試みると、エラーが発生し、オーバーライドの値が、**override** の変数に返されます。 必要ならば、もう一度命令を送れば、エラーは取り消され、校正が始まります。
リセット値	N/A
クエリに対する応答	N/A
解説	測定器の自己校正を実行する。 通常、EX1048 が連続して 60 分以上電源が入った状態になる前に、校正を実行するべきではありません。 実際、必要な時間が経過する前に校正をしようとするとエラーが発生します。 以下の説明にあるように、必要ならば、もう一度命令を送ると、エラーが取り消されます。 **注: 校正を実行する前に、測定器のロックを確保する必要があります。 ロックをしないで校正を実行しようとするとエラーが発生し、それを取り消すことはできません。*vtex1048_lock* の命令をご参照ください。** **注: 校正をする前に、電源の入った時間が必要なのは、測定器の測定精度を確保するためです。 この条件を無視すのは、動作環境が許す場合に限るべきです。 例として、測定器が十分温まっていて、瞬間的に電源を落としたり、再起動した場合などが考えられます。この条件が、故意に無視されたことを確認する意味で、アプリケーションソフトウェアで、ユーザーの対応が必要なようにすることを、強くお勧めします。** **注: 校正が無事に開始したときは、何パーセント完了したかを *vtex1048_self_cal_get_status* の命令でしレベルことができます。100% 完了する前に、次の命令を実行しないでください。**
例	

vtex1048_self_cal_is_stored

用途	不揮発性メモリ上の校正値の存在を尋ねる。
種類	クエリ
命令文法	vtex1048_self_cal_is_stored(**vi, &stored**)
パラメータ	**vi** = セッション ID **stored** = 校正値が不揮発性メモリーに格納されているかどうかを示す論理値。
リセット値	N/A
クエリに対する応答	**stored**: 0 または 1
解説	不揮発性メモリ上の校正値の存在を尋ねる。不揮発性メモリ上の校正値は、測定器の電源が入ったとき、または、リセットされたときに自動的に読み出されます。*vtex1048_self_cal_store* の命令で保存されます。
例	

vtex1048_self_cal_load

用途	不揮発性メモリ上の校正値を現在の校正値として読みこむ。
種類	イベント
命令文法	vtex1048_self_cal_load(**vi**)
パラメータ	**vi** = セッション ID
リセット値	N/A
クエリに対する応答	N/A
解説	不揮発性メモリ上の校正値を現在の校正値として読み込む。　この命令で、現在の校正値は自動的に上書きされますので、あらかじめ消去しておく必要はありません。　不揮発性メモリ上に校正値が保存されていないときにこの命令が送られると、エラーが発生します。
例	

vtex1048_self_cal_store

用途	現在の校正値を不揮発性メモリに保存する。
種類	イベント
命令文法	vtex1048_self_cal_store(**vi**)
パラメータ	**vi** = セッション ID
リセット値	N/A
クエリに対する応答	N/A
解説	現在の校正値を不揮発性メモリに保存すれ、測定器の電源が入ったとき、またはリセットされたときに現在の校正値として使われます。　現在の校正値が存在しないときにこの命令が送られるとエラーが発生します。　不揮発性メモリ上の校正値は工場出荷時の校正値の恒久的変更（取り消すことは可能ですが）を意味しますので、その存在を *vtex1048_self_cal_is_stored* の命令で尋ねることができます。
例	

vtex1048_set_arm_count

用途	アーム回数を設定する。
種類	設定
命令文法	vtex1048_set_arm_count(**vi**, **count**)
パラメータ	**vi** = セッション ID **count** = アーム回数の値。 1 から (2^31-1)までの整数値。
リセット値	1
クエリに対する応答	N/A
解説	アーム回数を設定する。この値はトリガの開始か、初期連続モードで、0 に達したとき自動的にリセットされます。

例	// set an arm count of 10
	vtex1048_set_arm_count(vi, 10);

vtex1048_set_arm_delay

用途	アーム遅延を設定する。
種類	設定
命令文法	vtex1048_set_arm_delay(**vi**, **arm_delay**)
パラメータ	**vi** = セッション ID **arm_delay** = アーム遅延を表す値（秒単位）　0 から 4294（71.5 分）で分解能 0.000001（1 µs）の値。
リセット値	0
クエリに対する応答	N/A
解説	アーム遅延を設定する。　アームイベントを検出してから、トリガモデルのトリガレイヤーに移行するまでの時間。

例	// set an arm delay of 5 ms
	vtex1048_set_arm_delay(vi, 0.005);

vtex1048_set_arm_infinite

用途	無限回アームを有効、または無効にする。
種類	設定
命令文法	vtex1048_set_arm_infinite(**vi**, **arminf**)
パラメータ	**vi** = セッション ID **arminf** = 無限回アームの有効状態を示す論理値
リセット値	0
クエリに対する応答	N/A
解説	無限回アームを有効、または無効にする。 この設定が有効の場合は、手動のアーム回数の設定はすべて無効になります。
例	

vtex1048_set_arm_source

用途	アームソースイベントを設定する。
種類	設定
命令文法	vtex1048_set_arm_source(**vi**, **vtb_masks[]**, **dio_masks[]**, **timer_enable**, **immediate**)
パラメータ	**vi** = セッション ID **vtb_masks[]** = ８チャンネルのトリガバスのアームイベントの有効状態を示す８ビットの整数値４個分の配列。値の順番は、上がりエッジ、下りエッジ、正のレベル、負のレベル。　0-255 (10 進)、または 0x00-0xFF(Hex)の値。　８ビットのうち MSB は VTB チャンネル 7,　LSB は VTB チャンネル 0 に対応。 **dio_masks[]** = ８チャンネルのデジタル IO ポートのアームイベントの有効状態を示す８ビットの整数値４個分の配列。値の順番は、上がりエッジ、下りエッジ、正のレベル、負のレベル。　８ビットのうち MSB は DIO チャンネル 7,　LSB は DIO チャンネル 0 に対応。 **timer_enable** = アームイベントとしてタイマーが有効かどうかを示す論理値。 **immediate** = アームイベントとして即時アームが有効かどうかを示す論理値。
リセット値	アームソースは即時に設定されます。
クエリに対する応答	N/A
解説	アームソースイベントを設定する。設定の値にかかわらず、ソフトウェアによるアームは常に有効です。

例	`// enable timer arm only`
	`ViUInt8 vtb_masks[4] = {0,0,0,0};`
	`ViUInt8 dio_masks[4] = {0,0,0,0};`
	`vtex1048_set_arm_source(vi, vtb_masks, dio_masks, 1, 0);`
	
	`// enable arm on a positive level on DIO channels 0-3 and a negative edge on VTB channel 6`
	`ViUInt8 vtb_masks[4] = {0,64,0,0};`
	`ViUInt8 dio_masks[4] = {0,0,0x0F,0};`
	`vtex1048_set_arm_source(vi, vtb_masks, dio_masks, 0, 0);`
	
	`// enable software arm only`
	`ViUInt8 vtb_masks[4] = {0,0,0,0};`
	`ViUInt8 dio_masks[4] = {0,0,0,0};`
	`vtex1048_set_arm_source(vi, vtb_masks, dio_masks, 0, 0);`

vtex1048_set_channel_conversion

用途	指定したチャンネルの工学単位(EU) 変換値を設定する。
種類	設定
命令文法	vtex1048_set_channel_conversion(**vi**, **channels[]**, **numChannels**, **eu_conv**)
パラメータ	**vi** = セッション ID **channels[]** =工学単位変換値を設定するチャンネルのリスト。 スキャンリストに含まれないチャンネル、またはスキャンリストの一部のチャンネルも可。 **numChannels** = チャンネルリストの配列の長さ。 1 から 48 までの整数。 **eu_conv** = 以下のいずれかの整数:
リセット値	**eu_conv** = すべてのチャンネルで VTEX1048_CONV_MV
クエリに対する応答	N/A
解説	指定したチャンネルの工学単位(EU) 変換値を設定する。

例	// channels 0-4 are E, channels 5-8 are T
	#define TYPE_E 0x04
	#define TYPE_T 0x03
	ViInt32 e_channels[5] = { 0, 1, 2, 3, 4 };
	vtex1048_set_channel_conversion(vi, e_channels, 5, TYPE_E);
	ViInt32 t_channels[4] = { 5, 6, 7, 8 };
	vtex1048_set_channel_conversion(vi, t_channels, 4, TYPE_T);

値	整数	関数
VTEX1048_CONV_MV	0	Voltage
VTEX1048_CONV_THERMO_TYPE_J	1	Type J
VTEX1048_CONV_THERMO_TYPE_K	2	Type K
VTEX1048_CONV_THERMO_TYPE_T	3	Type T
VTEX1048_CONV_THERMO_TYPE_E	4	Type E
VTEX1048_CONV_THERMO_TYPE_B	5	Type B
VTEX1048_CONV_THERMO_TYPE_S	6	Type S
VTEX1048_CONV_THERMO_TYPE_R	7	Type R
VTEX1048_CONV_THERMO_TYPE_N	8	Type N
VTEX1048_CONV_USER_DEF0	9	User-defined 0
VTEX1048_CONV_USER_DEF1	10	User-defined 1

vtex1048_set_dio_limit_event

用途	リミット評価をデジタルIOポートの動作と結合する。
種類	設定
命令文法	vtex1048_set_dio_limit_event(**vi**, **dio_channel**, **limit_masks[]**)
パラメータ	**vi** = セッション ID **dio_channel** = DIO のリミットイベントの状態を設定するチャンネル。 0 から 7 までの整数。 **limit_masks[]** =各チャンネルごとに、４種類のリミット条件をあらわす、４８個の４ビットの整数配列。 ４ビットのうち、順番は、リミットセット 0 の下限、リミットセット 0 の上限、リミットセット 1 の下限、リミットセット 1 の上限。 チャンネル 0 から 47 は配列の要素[0]から[47]に対応。
リセット値	**limit_masks[]** = すべての DIO チャンネルに対応するすべての配列の要素は 0
クエリに対する応答	N/A
解説	リミット評価をデジタルIOポートの動作と結合する。 通常の動作では、リミットに設定された DIO チャンネルは限界を超えたときに、low から high に移行します。 DIO チャンネルに論理 OR で複数のリミットを関係づけることができます つまり、４入力チャンネルのリミットに関連付けられた DIO チャンネルは、４つのどのチャンネルが限界を超えても、レベルが変化します。 複数の関連付けは、同じチャンネル、または、複数のチャンネルにまたがることも可能です。 通常、限界を超えたとき、関連付けられた DIO チャンネルは low から high に移行します。しかし、*vtex1048_set_dio_limit_event_invert* の命令で、high から low に移行するようにも設定できます。 関連付けられた、DIO チャンネルの通常の動作は、最新のリミットの状態で決まります。つまり、各スキャン毎に更新されます。 しかし、*vtex1048_set_dio_limit_event_latch* の命令で、ラッチ動作に指定することも可能です。 ラッチモードでは、クリアされた状態から、その後のリミットの状態に関わらず、次の、新たな測定の開始でクリアされるまで、最初のリミットによる変化が維持されます。

例	// link DIO channel 7 to set 1 lower limits on ch 12 and 14, set 0 upper limit on ch 1
	ViUInt8 limit_masks[48] =
	{0,4,0,0,0,0,0,0,0,0,0,2,0,2,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0};
	
	vtex1048_set_dio_limit_event(vi, 7, limit_masks);

vtex1048_set_dio_limit_event_invert

用途	リミットイベントに指定された DIO の反転動作を有効/無効にする。
種類	設定
命令文法	vtex1048_set_dio_limit_event_invert(**vi**, **dio_channel**, **invert**)
パラメータ	**vi** = セッション ID **dio_channel** = デジタル I/O ポートチャンネル。 0 から 7 までの整数。 **invert** = 反転モードで動作しているかどうかを表す論理値。
リセット値	**invert** = すべての DIO チャンネルで 0
クエリに対する応答	N/A
解説	リミットイベントに指定された DIO の反転動作を有効/無効にする。リミットを越えた場合、通常の DIO チャンネルは low から high に変わりますが、変転モードでは、high から low に変わります。
例	

vtex1048_set_dio_limit_event_latch

用途	リミットイベントに指定された DIO のラッチ動作を有効/無効にする。
種類	設定
命令文法	vtex1048_set_dio_limit_event_latch(**vi, dio_channel, latch**)
パラメータ	**vi** = セッション ID **dio_channel** = DIO ポートのチャンネル。 0 から 7 までの整数。 **latch** = 指定した DIO チャンネルがラッチモードかどうかを示す論理値。
リセット値	**latch** = すべての DIO チャンネルで 0。
クエリに対する応答	N/A
解説	リミットイベントに指定された DIO のラッチ動作を有効/無効にする。通常の動作では、DIO チャンネルは最後のリミットの状態を反映します。 つまり、各スキャンごとに交信されます。 ラッチモードではその後のリミットの状態に関係なく、クリアしたあとの最初の変化が、次の測定の最初にクリアされるまで、そのまま維持されます。
例	

vtex1048_set_dio_output

用途	有効な各DIOポートの出力レベルを設定する。
種類	設定
命令文法	vtex1048_set_dio_output(**vi**, **dio_out**)
パラメータ	**vi** = セッション ID **dio_out** = ８ビットポートのプログラム出力のステートをあらわす整数値。 ８ビットのフィールドのうち MSB は DIO のチャンネル７、LSB は DIO のチャンネル 0 に対応します。
リセット値	0
クエリに対する応答	N/A
解説	有効な各DIOポートの出力レベルを設定する。 *vtex1048_set_dio_output_enable* の命令で有効にされます。

例	// set DIO bit 7 (high) and DIO bit 6 (low)
	vtex1048_set_dio_output(vi, 0x80);
	// enable them as outputs
	vtex1048_set_dio_output_enable(vi, 0xC0);

vtex1048_set_dio_output_enable

用途	各DIOチャンネルの出力機能を有効、または無効にする。
種類	設定
命令文法	vtex1048_set_dio_output_enable(**vi**, **out_enable**)
パラメータ	**vi** = セッション ID **out_enable** = ８ビットポートの出力の有効状態をあらわす整数値。 0-255(十進)、0x00-0xFF(Hex)の整数。 ８ビットのフィールドのうち MSB は DIO のチャンネル７、LSB は DIO のチャンネル０に対応します。
リセット値	0
クエリに対する応答	N/A
解説	各DIOチャンネルの出力機能を有効、または無効にする。 各チャンネルの入力の機能は出力機能にかかわらず一定です。

例	// set DIO bit 7 (high) and DIO bit 6 (low)
	vtex1048_set_dio_output(vi, 0x80);
	// enable them as outputs
	vtex1048_set_dio_output_enable(vi, 0xC0);

vtex1048_set_dio_pulse

用途	DIOポートの指定したチャンネルから1 µs のパルスを発生させる。
種類	イベント
命令文法	vtex1048_set_dio_pulse(**vi**, **dio_pulse**)
パラメータ	**vi** = セッション ID **dio_pulse** = ８ビットポートのうちのパルス出力のチャンネルを表します。 0-255(十進)、0x00-0xFF(Hex)の整数。 ８ビットのフィールドのうち MSB は DIO のチャンネル７、LSB は DIO のチャンネル０に対応します。
リセット値	N/A
クエリに対する応答	N/A
解説	DIOポートの指定したチャンネルから1 µsのパルスを発生させる。 指定したチャンネルの出力が有効な場合に限りパルスが発生されます。 チャンネルの定常状態がhighにプログラムされていれば、負のパルスが、また、lowにプログラムされていれば正のパルスが発生します。
例	

	// set DIO bit 7 low and then pulse high
	vtex1048_set_dio_output(vi, 0x00);
	vtex1048_set_dio_output_enable(vi, 0x80);
	vtex1048_set_dio_pulse(vi, 0x80);

vtex1048_set_fifo_config

用途	FIFO メモリーのデータの書式とオーバーフロー動作を設定する。
種類	設定
命令文法	vtex1048_set_fifo_config(**vi, report_cjc, report_timestamp, report_celsius, blocking_mode**)
パラメータ	**vi** = セッション ID **report_cjc** =測定データに CJC 温度がレポートされるかどうかを示す論理値。この設定はデータの表示のみで、測定には影響しません。 CJC 温度は設定にかかわらず、各スキャンで測定されます。 CJC 温度は、**report_celsius** パラメータの値にかかわらず常に °C の単位であらわされます。 **report_timestamp** =測定データにチャンネルごとのタイムスタンプがレポートされるかどうかを示す論理値。 タイムスタンプはスキャンの開始時刻から、特定の測定までの時間間隔を 100 ns 単位であらわした時間です。 **report_celsius** =入力チャンネルのデータの単位が °C(真) か °F(偽)をあらわす論理値。 **blocking_mode** = スキャン中に最大容量に達したときの FIFO の動作を示す論理値。ブロッキングモードが有効な場合は、それ以降の測定値は排除され、バッファの内容が維持されます。 ブロッキングモードが無効な場合は、Circular (ループ) バッファとして働き、古いデータは新しいデータに上書きされます。
リセット値	**report_cjc** = 0 **report_timestamp** = 0 **report_celsius** = 1 **blocking_mode** = 0
クエリに対する応答	N/A
解説	FIFO メモリのデータの書式とオーバーフロー動作を設定する。 **注: report_cjc と report_timestamp パラメータの値にかかわらず, CJC 温度とタイムスタンプの変化分の情報は *vtex1048_read_fifo* の命令ではアクセスできません。これらの情報は、ストリーミングインターフェースと相当するパラメータによって得ることができます。**
例	

vtex1048_set_filt_freq

用途	指定したチャンネルのハードウェアフィルターの周波数を設定する
種類	設定
命令文法	vtex1048_set_filt_freq(**vi**, **channels[]**, **numChannels**, **filt_freq**)
パラメータ	**vi** = セッション ID **channels[]** = ハードウェアフィルタの周波数を設定するチャンネルリスト。　スキャンリストに含まれないチャンネル、またはスキャンリストの一部のチャンネルも可。 **numChannels** = チャンネルリストの配列の長さ。　1 から 48 までの整数。 **filt_freq** = 4.0 または 1000.0 (Hz の単位)
リセット値	**filt_freq** = 4.0 Hz for all channels
クエリに対する応答	N/A
解説	指定したチャンネルのハードウェアフィルタの周波数を設定する。

例	// set channels 0-4 for 4 Hz
	ViInt32 low_channels[5] = { 0, 1, 2, 3, 4 };
	vtex1048_set_filt_freq(vi, low_channels, 5, 4.0);
	
	// set channels 5-8 for 1 kHz
	ViInt32 high_channels[4] = { 5, 6, 7, 8 };
	vtex1048_set_filt_freq(vi, high_channels, 4, 1000.0);

vtex1048_set_init_cont

用途	初期連続モードを有効、または無効にする。
種類	設定
命令文法	vtex1048_set_init_cont(**vi**, **init_cont_mode**)
パラメータ	**vi** = セッション ID **init_cont_mode** = 初期連続モードが有効かどうかを示す論理値。
リセット値	0
クエリに対する応答	N/A
解説	初期連続モードを有効、または無効にする。初期連続モードでは新しいトリガ開始命令がなくても、トリガモデルを ARM レイヤーの入り口に戻します。
例	

vtex1048_set_limit_set0

用途	指定したチャンネルのリミットセット0の値を手動で設定する。
種類	設定
命令文法	vtex1048_set_limit_set0(**vi,channels[], numChannels, lower_limit, upper_limit**)
パラメータ	**vi** = セッション ID **channels[]** = リミットセット 0 の値を設定するチャンネルリスト。 スキャンリストに含まれないチャンネル、またはスキャンリストの一部のチャンネルも可。 **numChannels** = チャンネルリストの配列の長さ。 1 から 48 までの整数。 **lower_limit** = 下限の値。 特定のチャンネルの測定値が指定した下限の値よりも低いときにリミットが起こります。 正常に動作するためには、リミットの値はそのチャンネルの測定単位と同じ単位で指定してください。 **upper_limit** = 上限の値。特定のチャンネルの測定値が指定した上限の値よりも高いときにリミットが起こります。 正常に動作するためには、リミットの値はそのチャンネルの測定単位と同じ単位で指定してください。
リセット値	**lower_limit** = すべてのチャンネルで、-6.6e-2 **upper_limit** = すべてのチャンネルで、-6.6e-2
クエリに対する応答	N/A
解説	指定したチャンネルのリミットセット0の値を手動で設定する。 チャンネルのリミット値はスキャンリストと関係なく設定することができ、ひとつの命令で、複数のチャンネルを同一の値に設定することができます。 しかし、異なる値の組み合わせを設定するには、個別に命令を実行する必要があります。 リミットセット 0 の初期設定値は、チャンネルの工学単位変換係数と工学単位の設定から、自動的に決められます。 手動のリミット操作が指定されている場合は、*vtex1048_set_limit_set0_manual* の命令で、リミット値をユーザーが定義することができます。 手動のリミットコントロールが有効になっていない場合は、この命令を実行するとエラーが発生し、設定値は無視されます。 正常に動作するためには、リミット値はチャンネルの測定単位と同じ単位で設定する必要があります。 一度設定されると、そのあとの工学単位の変換値や工学単位の入力で、自動的に更新されることはありません。

例	`// set channels 0-4 to manual limits of 0 and 100`
	`ViInt32 e_channels[5] = { 0, 1, 2, 3, 4 };`
	`vtex1048_set_limit_set0_manual(vi, e_channels, 5, 1);`
	`vtex1048_set_limit_set0(vi, e_channels, 5, 0, 100);`

vtex1048_set_limit_set0_manual

用途	指定したチャンネルのリミットセット 0 の手動入力を有効、無効にする。
種類	設定
命令文法	vtex1048_set_limit_set0_manual(**vi, channels[], numChannels, manual**)
パラメータ	**vi** = セッション ID **channels[]** = 手動で、リミット値の入力を有効にするチャンネルリスト。スキャンリストに含まれないチャンネル、またはスキャンリストの一部のチャンネルも可。 **numChannels** = チャンネルリストの配列の長さ。 1 から 48 までの整数。 **manual** =手動入力が有効かどうかを表す論理値。
リセット値	すべてのチャンネルで 0
クエリに対する応答	N/A
解説	指定したチャンネルのリミットセット 0 の手動入力を有効、無効にする。 チャンネルの手動コントロールはスキャンリストと関係なく設定することができ、1 つの命令で、複数のチャンネルを同時に設定することができます。 しかし、チャンネルごとに異なる設定するには、個別の命令を実行する必要があります。 リミットセット 0 の初期設定値は、チャンネルの工学単位変換係数と工学単位の設定から、自動的に決められます。 手動のリミット操作を希望する場合は、あらかじめこの命令で、リミット操作を有効にする必要があります。 あるチャンネルの、手動のリミット操作を有効から、無効にかえると、リミットセット 0 の値が、現在の工学単位変換係数と、工学単位から自動的に設定されます。
例	

vtex1048_set_limit_set1

用途	指定したチャンネルのリミットセット 1 の値を設定する。
種類	設定
命令文法	vtex1048_set_limit_set1(**vi**, **channels[]**, **numChannels**, **lower_limit**, **upper_limit**)
パラメータ	**vi** = セッション ID **channels[]** = リミットセット 1 の値を設定するチャンネルリスト。 スキャンリストに含まれないチャンネル、またはスキャンリストの一部のチャンネルも可。 **numChannels** = チャンネルリストの配列の長さ。 1 から 48 までの整数。 **lower_limit** = 下限の値。 特定のチャンネルの測定値が指定した下限の値よりも低いときにリミットが起こります。 正常に動作するためには、リミットの値はそのチャンネルの測定単位と同じ単位で指定してください。 **upper_limit** = 上限の値。特定のチャンネルの測定値が指定した上限の値よりも高いときにリミットが起こります。 正常に動作するためには、リミットの値はそのチャンネルの測定単位と同じ単位で指定してください。
リセット値	**lower_limit** = すべてのチャンネルで-3e+38 **upper_limit** = すべてのチャンネルで 3e+38
クエリに対する応答	N/A
解説	指定したチャンネルのリミットセット 1 の値を設定する。 チャンネルのリミット値はスキャンリストと関係なく設定することができ、ひとつの命令で、複数のチャンネルを同一の値に設定することができます。 しかし、異なる値の組み合わせを設定するには、個別の命令を実行する必要があります。 リミットセット 0 と異なり、リミットセット 1 は常に手動で設定されます。 正常に動作するためには、リミット値はチャンネルの測定単位と同じ単位で設定する必要があります。 一度設定されると、そのあとの工学単位の変換値や工学単位の入力で、自動的に更新されることはありません。
例	

vtex1048_set_scanlist

用途	測定のスキャンリストを設定する。
種類	設定
命令文法	vtex1048_set_scanlist(**vi**, **channels[]**, **numChannels**)
パラメータ	**vi** = セッション ID **channels[]** = スキャンリストに含めるチャンネルのリスト。　各要素は 0-47 の固有の整数。　要素の順番はデータの測定の順番に対応。 **numChannels** = チャンネルリストの配列の長さ。　1 から 48 までの整数。
リセット値	**channels[0]** = 0 **numChannels** = 1
クエリに対する応答	N/A
解説	測定のスキャンリストを設定する。 有効なスキャンリストの条件は • 最低 1 チャンネル。 • 最高 48 チャンネル。 • チャンネルの重複は無し。

例	`// sequential order five-channel scan`
	`ViInt32 channels[5] = { 0, 1, 2, 3, 4 };`
	`vtex1048_set_scanlist(vi, channels, 5);`
	
	`// reverse order five-channel scan`
	`ViInt32 channels[5] = { 4, 3, 2, 1, 0 };`
	`vtex1048_set_scanlist(vi, channels, 5);`

vtex1048_set_trig_source_timer

用途	タイマートリガと時間間隔を設定する。
種類	設定
命令文法	vtex1048_set_trig_source_timer(**vi**, **trig_timer**)
パラメータ	**vi** = セッション ID **trig_timer** = タイマーイベントの時間間隔(秒単位)。 0.001 (1 ms)から 4294 (71.5 minutes) までで、分解能 0.000001 (1 µs) の値。
リセット値	N/A
クエリに対する応答	N/A
解説	タイマートリガと時間間隔を設定する。 これは、*vtex1048_set_trigger_source* と *vtex1048_set_trigger_timer* の命令を複合した命令。
例	

vtex1048_set_trigger_count

用途	トリガ回数を設定する。
種類	設定
命令文法	vtex1048_set_trigger_count(**vi**, **count**)
パラメータ	**vi** = セッション ID **count** = トリガー回数。1 から(2^{31}-1)間での整数値。
リセット値	1
クエリに対する応答	N/A
解説	トリガ回数を設定する。　この値は、各アームイベントごとにリセットされる。

例	// set a trigger count of 10
	vtex1048_set_trigger_count(vi, 10);

vtex1048_set_trigger_delay

用途	トリガ遅延を設定する。
種類	設定
命令文法	vtex1048_set_trigger_delay(**vi**, **trig_delay**)
パラメータ	**vi** = セッション ID **trig_delay** = 秒単位のトリガー遅延時間。　値の範囲は 0.000001(1 µs)刻み 0 から 4294 (71.5 分)。
リセット値	0
クエリに対する応答	N/A
解説	トリガ遅延を設定する。トリガイベントの検出からスキャンリストを実行するまでの遅延時間。

例	// set a trigger delay of 5 ms
	vtex1048_set_trigger_delay(vi, 0.005);

vtex1048_set_trigger_infinite

用途	無限回トリガを有効、または無効にする。
種類	設定
命令文法	vtex1048_set_trigger_infinite(**vi**, **triginf**)
パラメータ	**vi** = セッション ID **triginf** =無限回トリガーが有効かどうかを示す論理値。
リセット値	0
クエリに対する応答	N/A
解説	無限回トリガを有効、または無効にする。無限回トリガの場合は、手動トリガ回数の設定は影響しません。
例	

vtex1048_set_trigger_source

用途	トリガソースイベントを設定する。
種類	設定
命令文法	vtex1048_set_trigger_source(**vi**, **vtb_masks[]**, **dio_masks[]**, **timer_enable**, **immediate**)
パラメータ	**vi** = セッション ID **vtb_masks[]** = ８チャンネルトリガバスからのトリガーイベントの有効状態を示す８ビット整数４個分の配列。数値の順番は：上りエッジ、下りエッジ、正のレベル、負のレベル。　８ビットのフィールドのうち MSB は VTB チャンネル７、LSB は VTB チャンネル 0 に対応。 **dio_masks[]** = ８チャンネルの DIO ポートのトリガイベントの有効状態を示す８ビットの整数値４個分の配列。　数値の順番は：上りエッジ、下りエッジ、正のレベル、負のレベル。　８ビットのフィールドのうち MSB は DIO チャンネル７、LSB は DIO チャンネル 0 に対応。 **timer_enable** = タイマートリガが有効かどうかを示す論理値。 **immediate** = トリガイベントとして即時トリガーが有効かどうかを示す論理値。
リセット値	タイマー
クエリに対する応答	N/A
解説	トリガソースイベントを設定する。　設定値にかかわらず、ソフトウェアのトリガは常に有効です。

例	`// enable timer trigger only`
	`ViUInt8 vtb_masks[4] = {0,0,0,0};`
	`ViUInt8 dio_masks[4] = {0,0,0,0};`
	`vtex1048_set_trigger_source(vi, vtb_masks, dio_masks, 1, 0);`
	
	`// enable trigger on a positive level on DIO channels 0-3 and a negative edge on VTB channel 6`
	`ViUInt8 vtb_masks[4] = {0,64,0,0};`
	`ViUInt8 dio_masks[4] = {0,0,0x0F,0};`
	`vtex1048_set_trigger_source(vi, vtb_masks, dio_masks, 0, 0);`
	
	`// enable software trigger only`
	`ViUInt8 vtb_masks[4] = {0,0,0,0};`
	`ViUInt8 dio_masks[4] = {0,0,0,0};`
	`vtex1048_set_trigger_source(vi, vtb_masks, dio_masks, 0, 0);`

vtex1048_set_trigger_timer

用途	タイマーソースイベントの時間間隔を設定する。
種類	設定
命令文法	vtex1048_set_trigger_timer(**vi**, **trig_timer**)
パラメータ	**vi** = セッション ID **trig_timer** = タイマーイベントの秒単位の時間間隔 0.000001 (1 µs)刻みで 0.001 (1 ms) から 4294 (71.5 分) の値。
リセット値	0.1 (100 ms)
クエリに対する応答	N/A
解説	タイマーソースイベントの時間間隔を設定する。　アームとトリガーイベントの両方に同じ値が使われます。

例	// set a timer interval of 50 ms
	vtex1048_set_trigger_timer(vi, 0.05);

vtex1048_set_user_cjc_enable

用途	指定したチャンネルのユーザー定義CJC温度を有効または無効にする。.
種類	設定
命令文法	vtex1048_set_user_cjc_enable(**vi**, **channels[]**, **numChannels**, **usercjc**)
パラメータ	**vi** = セッション ID **channels[]** = CJC 温度の有効状態を設定するチャンネルリスト。 スキャンリストに含まれないチャンネル、またはスキャンリストの一部のチャンネルも可。 **numChannels** = チャンネルリストの配列の長さ。 1 から 48 までの整数。 **usercjc** = ユーザー定義の CJC 温度が有効かどうかを示す論理値。
リセット値	**usercjc** = すべてのチャンネルで 0
クエリに対する応答	N/A
解説	指定したチャンネルのユーザー定義CJC温度を有効または無効にする。 これが設定されていると、熱伝対の計算に内部での測定値の代わりに、ユーザーが定義したCJC温度が使われます。 各チャンネル毎に、外部 CJC の値と、有効か無効かを個別に設定できます。 つまり、値を設定しても自動的に使用されるわけではなく、また、その使用を無効にしても、以前に設定した値がクリアされるわけではありません。 CJC 温度の設定は *vtex1048_set_user_cjc_temp* の命令で行います。 **注: この命令は、熱伝対冷接点が EX1048 の外部に設定されているときのみ使われます。この機能は注意して使用する必要があります。 測定データには、外部の CJC を使って計算されたか、または内部のものかの情報は含まれていません。**

例	// CJC for channels 0-4 are held externally at 0.2 C
	ViInt32 ext_channels[5] = { 0, 1, 2, 3, 4 };
	vtex1048_set_user_cjc_temp(vi, ext_channels, 5, 0.2);
	vtex1048_set_user_cjc_enable(vi, ext_channels, 5, 1);

vtex1048_set_user_cjc_temp

用途	指定したチャンネルのユーザー定義CJC温度を設定する。
種類	設定
命令文法	vtex1048_set_user_cjc_temp(**vi, channels[], numChannels, cjc_temp**)
パラメータ	**vi** = セッション ID **channels[]** = CJC 温度を設定するチャンネルリスト。 スキャンリストに含まれないチャンネル、またはスキャンリストの一部のチャンネルも可。 **numChannels** = チャンネルリストの配列の長さ。 1 から 48 までの整数。 **cjc_temp** = CJC 温度。 チャンネルの測定単位にかかわらず °C の単位で設定。
リセット値	**cjc_temp** = すべてのチャンネルで 0
クエリに対する応答	N/A
解説	指定したチャンネルのユーザー定義CJC温度を設定する。これが設定されていると、熱伝対の計算に内部での測定値の変わりに、ユーザーが定義したCJC温度が使われます。各チャンネルごとに、外部 CJC の値と、有効か無効かを独立して設定できます。 つまり、値を設定しても自動的に使用されるわけではなく、またその使用を無効にしても、以前に設定した値がクリアされるわけではありません。 CJC を有効にするのは、*vtex1048_set_user_cjc_enable* の命令で行います。 **注: この命令は、熱伝対冷接点が EX1048 の外部に設定されているときのみ使われます。この機能は注意して使用する必要があります。 測定データには、外部の CJC を使って計算されたか、または内部のものかの情報は含まれていません。**

例	`// CJC for channels 0-4 are held externally at 0.2 C`
	`ViInt32 ext_channels[5] = { 0, 1, 2, 3, 4 };`
	`vtex1048_set_user_cjc_temp(vi, ext_channels, 5, 0.2);`
	`vtex1048_set_user_cjc_enable(vi, ext_channels, 5, 1);`

vtex1048_set_user_conversion

用途	ユーザー定義の変換多項式を設定する。
種類	設定
命令文法	vtex1048_set_user_conversion(**vi, eu_conv, fwdcoeff[], numFwd, invcoeff[], numInv**)
パラメータ	**vi** = セッション ID **eu_conv** = 定義する多項式セット 9（ユーザー0）または 10（ユーザー1） **fwdcoeff[]** = 正変換多項式の係数。 係数 c_0 から c_{12} までは配列要素 [0] から [12] に対応。 **numFwd** = 正変換係数の配列の長さ。 1 から 13 までの整数。 **invcoeff[]** = 逆変換多項式の係数。 係数 d_0 から d_{12} までは配列要素 [0] から [12]に対応。 **numInv** =逆変換係数の配列の長さ。 1 から 13 までの整数。
リセット値	**fwdcoeff[]** = 両方の多項式のすべての配列要素は 0 **invcoeff[]** = 両方の多項式のすべての配列要素は 0
クエリに対する応答	N/A
解説	ユーザー定義の変換多項式を設定する。 正変換多項式は、CJC 温度を補正された冷接点の電圧にに変換するのに使われます: $$E = c_0 + c_1 * t^1 + c_2 * t^2 + \ldots + c_{12} * t^{12}$$ ここで、 E 電圧, t は °C, そして c_o – c_{12} は係数。 逆変換多項式は、補正された入力電圧を以下の式で温度に変換するのに使われます: $$t = d_0 + d_1 * E^1 + d_2 * E^2 + \ldots + d_{12} * E^{12}$$ ここで、 E 電圧, t は °C, そして c_o – c_{12} は係数。 未定義係数は自動的に 0 に設定されます。 **注: 係数を設定しただけでは、自動的に有効にはなりません。 有効にするには、User0 か User1 の EU 変換を *vtex1048_set_channel_conversion* の命令で指定してください。**
例	

vtex1048_set_vtb_output

用途	有効なトリガバスの各チャンネルの出力レベルを設定する。
種類	設定
命令文法	vtex1048_set_vtb_output(**vi**, **vtb_out**)
パラメータ	**vi** = セッション ID **vtb_out** = 希望の 8 ビットポートの状態を表す整数値。 値の範囲は、0-255 (十進)または0x00-0xFF (Hex)。 8 ビットのうち MSB は VTB チャンネル 7, LSB は VTB チャンネル 0 に対応。
リセット値	0
クエリに対する応答	N/A
解説	有効なトリガバスの各チャンネルの出力レベルを設定する。 *vtex1048_set_vtb_output_enable*の命令で有効になります。

例	// set VTB bit 7 (high) and VTB bit 6 (low)
	vtex1048_set_vtb_output(vi, 0x80);
	// enable them as outputs
	vtex1048_set_vtb_output_enable(vi, 0xC0);

vtex1048_set_vtb_output_enable

用途	トリガーバスの各チャンネルの出力機能を有効または無効にする。
種類	設定
命令文法	vtex1048_set_vtb_output_enable(**vi**, **out_enable**)
パラメータ	**vi** = セッション ID **out_enable** =希望の 8 ビットポートの状態を表す整数値。 値の範囲は、0-255 (十進)または0x00-0xFF (Hex)。 8 ビットのうち MSB は VTB チャンネル 7、 LSB は VTB チャンネル 0 に対応。
リセット値	0
クエリに対する応答	N/A
解説	トリガーバスの各チャンネルの出力機能を有効または無効にする。 出力の機能にかかわらず、各チャンネルの入力の機能は一定です。

例	// set VTB bit 7 (high) and VTB bit 6 (low)
	vtex1048_set_vtb_output(vi, 0x80);
	// enable them as outputs
	vtex1048_set_vtb_output_enable(vi, 0xC0);

vtex1048_set_vtb_pulse

用途	トリガーバスの指定したチャンネルに 1 µs のパルスを発生する
種類	イベント
命令文法	vtex1048_set_vtb_pulse(**vi**, **vtb_pulse**)
パラメータ	**vi** = セッション ID **vtb_pulse** = パルスを出力する8 ビットポートのチャンネル。 値の範囲は、0-255 (十進)または0x00-0xFF (Hex)の整数。 8 ビットのうち MSB は VTB チャンネル 7、 LSB は VTB チャンネル 0 に対応。
リセット値	N/A
クエリに対する応答	N/A
解説	トリガーバスの指定したチャンネルに 1 µs のパルスを発生する。指定したチャンネルの出力が有効の場合に限りパルスが出力されます。チャンネルの定常値がhighの場合は負のパルス、lowのばあいは正のパルスが出力されます。

例	// set VTB bit 7 low and then pulse high
	vtex1048_set_vtb_output(vi, 0x00);
	vtex1048_set_vtb_output_enable(vi, 0x80);
	vtex1048_set_vtb_pulse(vi, 0x80);

vtex1048_soft_arm

用途	ソフトウェアアームを実行する。
種類	イベント
命令文法	vtex1048_soft_arm(**vi**)
パラメータ	**vi** = セッション ID
リセット値	N/A
クエリに対する応答	N/A
解説	ソフトウェアアームを実行する。　アームソースの状態にかかわらず、ソフトウェアアームは常に有効です。
例	

vtex1048_soft_trigger

用途	ソフトウェアトリガを実行する。
種類	イベント
命令文法	vtex1048_soft_trigger(**vi**)
パラメータ	**vi** = セッション ID
リセット値	N/A
クエリに対する応答	N/A
解説	ソフトウェアトリガを実行する。　トリガソースの状態にかかわらず、ソフトウェアトリガは常に有効です。
例	

vtex1048_unlock

用途	測定器のロックを解除する。
種類	イベント
命令文法	vtex1048_unlock(**vi**)
パラメータ	**vi** = セッション ID
リセット値	N/A
クエリに対する応答	N/A
解説	測定器のロックを解除する。この命令は、ロックを取得したホスト IP アドレスからしか実行できません。　他のユーザーのロックを解除するには、*vtex1048_break_lock* の命令を使う必要があります。 *vtex1048_reset* は、ロックの状態に影響しないので、この命令で、ロックを解除することはできません。
例	

第 8 章

動作理論

はじめに

図 8-1のブロック図は本製品のアナログ回路の主要構成部品を図解しています。以下、各ブロックを説明します。

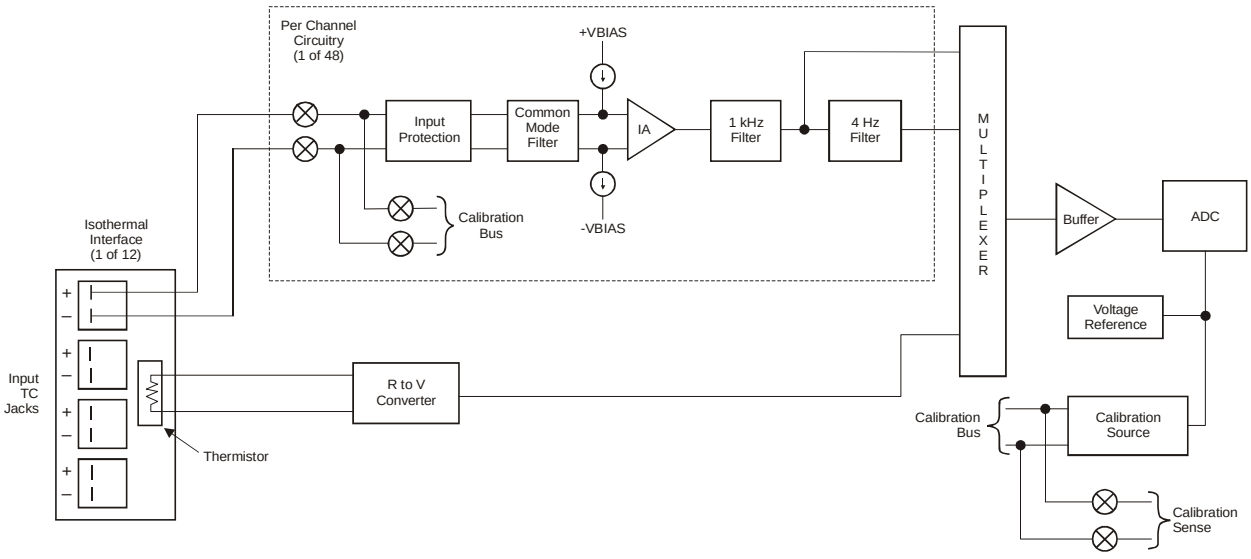


図8-1: EX1048 アナログ回路ブロック図

シグナルコンディション回路

各入力チャンネルは “Per Channel Circuitry”と表示されたブロックでシグナルコンディショニングされます。このブロックの中心となるのが差動入力信号の増幅とコモンモード入力信号の除去をする計装増幅回路(IA)です。計装増幅回路は高いコモンモード除去比を誇ります（特に dc、(50/60) Hz 帯において）が、干渉周波数の上昇と共にその特性は下がります。また、計装増幅回路は特に高い信号周波数において直流分を移動させてしまうという特性をもっています。特に問題となるのは、この効果はその後にくるフィルタでは減衰できないということです。そのため、本製品では計装増幅回路の前にコモンモードフィルタを置くことで事前に高周波信号を減衰させます。こうする事によって dc 整流のを防ぎ、すべての周波数帯において高レベルのコモンモード除去比（CMRR）特性を実現しています。

熱電対断線検出機能は各入力パスのナノアンプ級の微弱電圧電源によって提供されます。これらの電源は、測定精度に影響がでない程度に出力の小さいものですが、断線状態において計装増幅回路の高干渉入力を飽和させるだけの十分な出力を持っています。これらの電源は常に入力パスに繋がっている為、入力の監視は連続的に行われます。この為、断線が断続的なものでも検出できます。また、両入力パスに電源を備えているので、片方の入力リードが断線して、もう片方がアースに繋がっている場合でも検出します。

ノイズ性能の最適化の為、計装増幅回路の出力は 2 つのアナログフィルタを通されます。ユーザーは、この測定されるフィルタ出力をチャンネル毎に選択できます。テストアプリケーションにおいて、シグナルコンディションパスの帯域を接続されたセンサの特性に合わせることを可能にする 2 フィルタの選択は強力な武器になります。ここで特記することは、ここで利用されているフィルタは、連続的に接続された 2 つの独立した固定フィルタであるということです。対照的な方法としては、ユーザー選択に応じて特性の変化するフィルタを 1 つ備えるという方法があります。この方法の欠点は、フィルタの特性を変えたとき、有効なデータが取れるまで、一定時間の遅延を内在しているということです。これは、測定システムに不必要に大きな遅延を強要することになります。EX1048 では、2 種類のフィルタを使用しているため、遅延の心配がありません。

冷接点補償

冷接点補償は計 12 のサーミスタが各 4 つの恒温入力部を監視することによって達成されています。サーミスタは電気抵抗が入力温度を表す抵抗体のことです。スキャンの始めに各サーミスタの電気抵抗は対応電圧に変換され、今度はサーミスタの Steinhart-Hart 係数を利用して温度値に変換されます。この温度値より各熱電対特有の補償電圧が生成され、対応するチャンネルの入力電圧に加算されます。

校正

EX1048 の校正は完全校正と自己校正の二つの形を取ります。どちらにおいても、入力シグナルコンディションパスは入力端子の接続を断ち、内蔵校正ソースで駆動する校正バスに接続されます。複数の校正ソースポイントにおいてコンディションパスを測定することによって、各パスの電圧ゲインとオフセットが計算されます。完全校正の場合は、さらに校正ソースを精密電圧計で測定する必要があります。従って、自己校正は完全校正のサブセットだと言うこともできます。EX1048 は、入力切断機能を内蔵していますので、校正をする為にユーザーが実際に入力接続を切る必要はありません。

熱電対計算

EX1048 では二種類の熱電対特有の計算を行います。一つ目は、冷接点補償温度の補償電圧への変換とその補償電圧の測定された入力電圧値への加算。二つ目は、この合計電圧の最終熱電対温度への変換作業です。精度を最大限に高めるため、両計算は参照表や区分一次近似法でなく、NIST ITS-90 熱電対データベースの全次数多項式、また係数を使っています。

総索引

A

ARM イベント ..... *トリガ参照*
AutoIP.....24, 54

C

CJC ..... *冷接点補償参照*
CMRR.....17, 28

D

DHCP サーバ.....23, 54
DIO リミットイベント.....34, 50, 51, 72

F

FIFO
　カウントの取得.....48, 80
　データの書き込み.....38
　データの読み出し.....40, 48, 80
　初期化.....46, 56, 62, 78
　設定.....49, 69

I

IO メニュー.....49

L

LED.....33
LXI トリガバス.....16

M

MAC アドレス.....24, 54

N

NIST ITS-90 熱電対仕様表.....19, 27, 40

O

OTC 検出 ..... *熱電対断線検出参照*

T

TRIG イベント.....42

V

vtex1048_abort.....93
vtex1048_break_lock.....94
vtex1048_check_lock.....95
vtex1048_close.....96
vtex1048_get_accum_limit_status.....97
vtex1048_get_arm_count.....98
vtex1048_get_arm_delay.....99
vtex1048_get_arm_infinite.....100
vtex1048_get_arm_source.....101
vtex1048_get_channel_conversion.....102
vtex1048_get_dio_input.....103
vtex1048_get_dio_limit_event.....104
vtex1048_get_dio_limit_event_invert.....105
vtex1048_get_dio_limit_event_latch.....106
vtex1048_get_dio_output.....107
vtex1048_get_dio_output_enable.....108
vtex1048_get_fifo_config.....109
vtex1048_get_fifo_count.....110
vtex1048_get_filt_freq.....111
vtex1048_get_init_cont.....112
vtex1048_get_limit_set0.....113
vtex1048_get_limit_set0_manual.....114
vtex1048_get_limit_set1.....115
vtex1048_get_scanlist.....116
vtex1048_get_trigger_count.....117
vtex1048_get_trigger_delay.....118
vtex1048_get_trigger_infinite.....119
vtex1048_get_trigger_source.....120
vtex1048_get_trigger_timer.....121
vtex1048_get_user_cjc_enable.....122
vtex1048_get_user_cjc_temp.....123
vtex1048_get_user_conversion.....124
vtex1048_get_vtb_input.....125
vtex1048_get_vtb_output.....126
vtex1048_get_vtb_output_enable.....127
vtex1048_init.....128
vtex1048_init_imm.....129
vtex1048_lock.....130
vtex1048_read_fifo.....131
vtex1048_reset.....132
vtex1048_reset_fifo.....133
vtex1048_reset_trigger_arm.....134
vtex1048_revisionQuery.....135
vtex1048_self_cal_clear.....136
vtex1048_self_cal_clear_stored.....137
vtex1048_self_cal_get_status.....138
vtex1048_self_cal_init.....139
vtex1048_self_cal_is_stored.....140
vtex1048_self_cal_load.....141
vtex1048_self_cal_store.....142
vtex1048_set_arm_count.....143
vtex1048_set_arm_delay.....144
vtex1048_set_arm_infinite.....145
vtex1048_set_arm_source.....146
vtex1048_set_channel_conversion.....147
vtex1048_set_dio_limit_event.....148
vtex1048_set_dio_limit_event_invert.....149
vtex1048_set_dio_limit_event_latch.....150
vtex1048_set_dio_output.....151
vtex1048_set_dio_output_enable.....152
vtex1048_set_dio_pulse.....153
vtex1048_set_fifo_config.....154
vtex1048_set_filt_freq.....155
vtex1048_set_init_cont.....156
vtex1048_set_limit_set0.....157
vtex1048_set_limit_set0_manual.....158
vtex1048_set_limit_set1.....159
vtex1048_set_scanlist.....160
vtex1048_set_trig_source_timer.....161
vtex1048_set_trigger_count.....162
vtex1048_set_trigger_delay.....163

vtex1048_set_trigger_infinite 164
vtex1048_set_trigger_source 165
vtex1048_set_trigger_timer 166
vtex1048_set_user_cjc_enable 167
vtex1048_set_user_cjc_temp 168
vtex1048_set_user_conversion 169
vtex1048_set_vtb_output 170
vtex1048_set_vtb_output_enable 171
vtex1048_set_vtb_pulse 172
vtex1048_soft_arm 173
vtex1048_soft_trigger 174
vtex1048_unlock 175
VXI-11 デバイス検出 24, 55

ウ

ウェブインターフェイス 16
ウォームアップ 18, 23, 32, 53, 63

コ

コモンモード入力域 17
コモンモード入力範囲 28
コモンモード除去比 ..... CMRR *参照*

サ

サンプリングレート 15, 17, 22, 31, 32, 43

ス

スキャンリスト
  タイミング 32
  メニュー 47
  設定 31, 47, 66

タ

タイムスタンプ 38, 49, 69

デ

データ
  データ形式 38, 49, 69
  ファイルとしてダウンロード 48
  メニュー 48
データ収集 38, 46, 78
データ読み出し 40, 48
デジタル I/O 15, 33, 34, 36, 38, 42, 43, 49, 50
デバイスメニュー 52

ト

トリガ 38, 41, 46, 73
トリガバス(VTB) 16, 36, 37, 38, 42, 43, 50, 74, 76
トリガメニュー 46

ネ

ネットワーク設定 23, 54

ノ

ノイズ性能 22, 26, 28, 31, 43

フ

ファームウェア
  更新 55
フィルタ 14, 28, 48, 68

ブ

ブロッキングモード 49, 69

メ

メモリ ..... FIFO *参照*

ユ

ユーザー定義変換式 27, 40, 57, 67

リ

リセット
  デバイス 56, 62
  ネットワーク設定 24, 55
  ボタン 24, 55

ロ

ロック 38, 56, 64
ロックメニュー 56

仕

仕様 17, 19, 32

入

入力保護 17, 28
入力端子 15, 17, 20, 25

冷

冷接点補償 20, 29, 49, 57, 67, 178

初

初期設定 61

単

単位 ..... 温度単位 *参照*

命

命令リスト 89

外

外形寸法 17

工

工学単位(EU)変換 27, 47, 66

時

時間設定 25, 55

温

温度単位 29, 49, 69
温度精度 .....使用表*参照*

測

測定範囲 14, 28

熱

熱伝対断線検出 14
熱電対断線検出 33, 178
熱電対線 .....配線*参照*
熱電対計算 27, 40, 178

特

特徴 14

自

自己校正.....15, 19, 20, 32, 33, 53, 63, 178
自己校正.....32

配

配線.....25

限

限界値.....33, 69
限界値メニュー.....50

電

電圧精度.....使用表*参照*

高

高度設定.....57